iOS Application Development

Session 201

Jeremy Foo
iOS Neckbeard

iOS Development

Overview

- Paradigm
- Application Life Cycle
- iOS Model View Controller
- Events
- Resources
- Performance

Paradigm

How things fit together

MVC

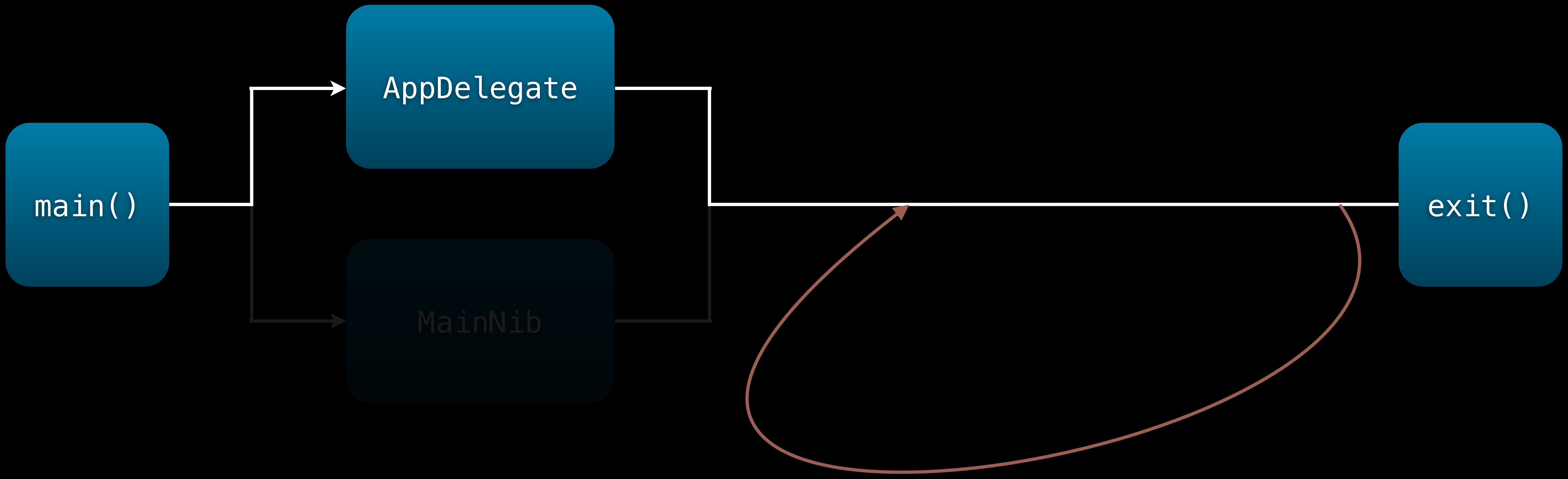
Event Driven

Application Life Cycle

Performance

Application Life Cycle

How life starts, progresses and ends



NSRunLoop

Multitasking

Events

main.m

Where everything starts

```
#import <UIKit/UIKit.h>

#import "AppDelegate.h"

int main(int argc, char *argv[])
{
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([AppDelegate class]));
    }
}
```

main.m

Where everything starts

- Same `main` implementation for all C programs
- Starts autorelease pool
- Start iOS ObjC Lifecycle using AppDelegate

Application Delegate

Application Entry Point

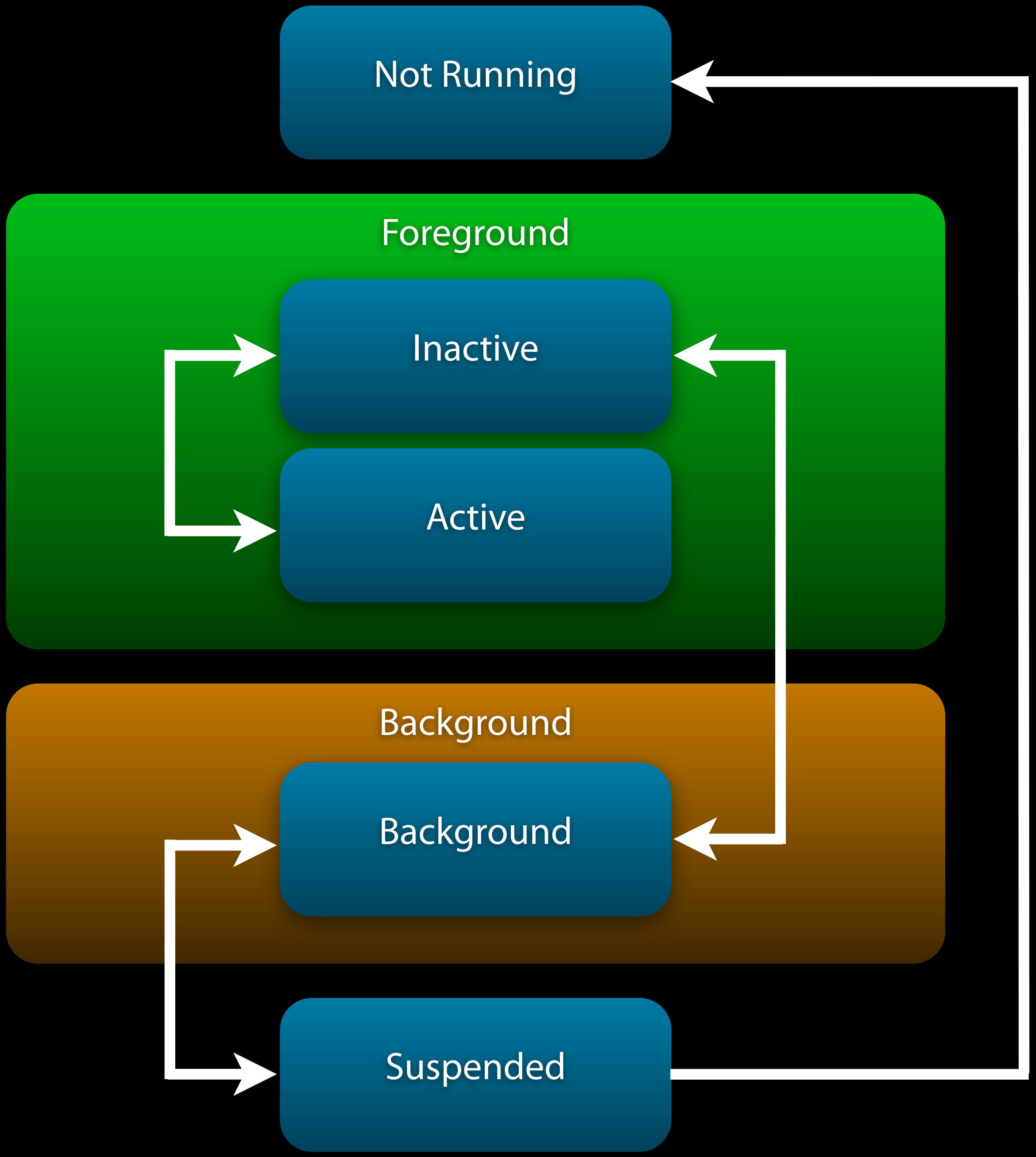
- Principal iOS application entry point
- Deals with Application specific call backs
- Some call backs duplicated as NSNotifications
- Usually sets up the initial view in the main screen UIWindow
- http://developer.apple.com/library/ios/#DOCUMENTATION/UIKit/Reference/UIApplicationDelegate_Protocol/Reference/Reference.html

Application Life Cycle
Multitasking states

- Multitasking States
  - Not Running
  - Inactive
  - Active
  - Background
  - Suspended

Application Life Cycle

Multitasking states



Application Life Cycle

Multitasking states

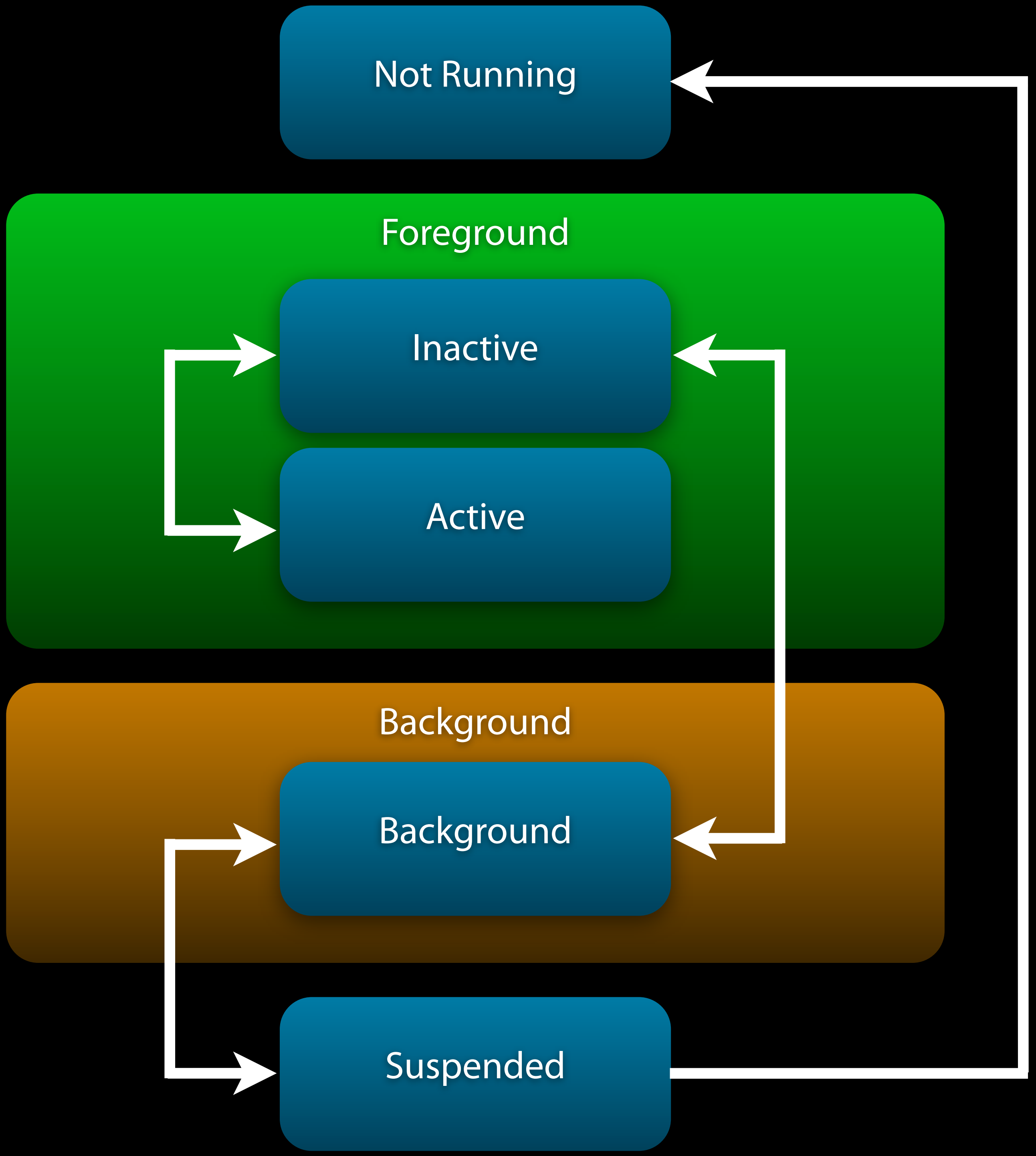
`application:DidFinishLaunchingWithOptions:`

`applicationWillEnterForeground`
`application:WillResignActive:`

`application:DidBecomeActive:`

`applicationDidEnterBackground`

`applicationWillTerminate:`

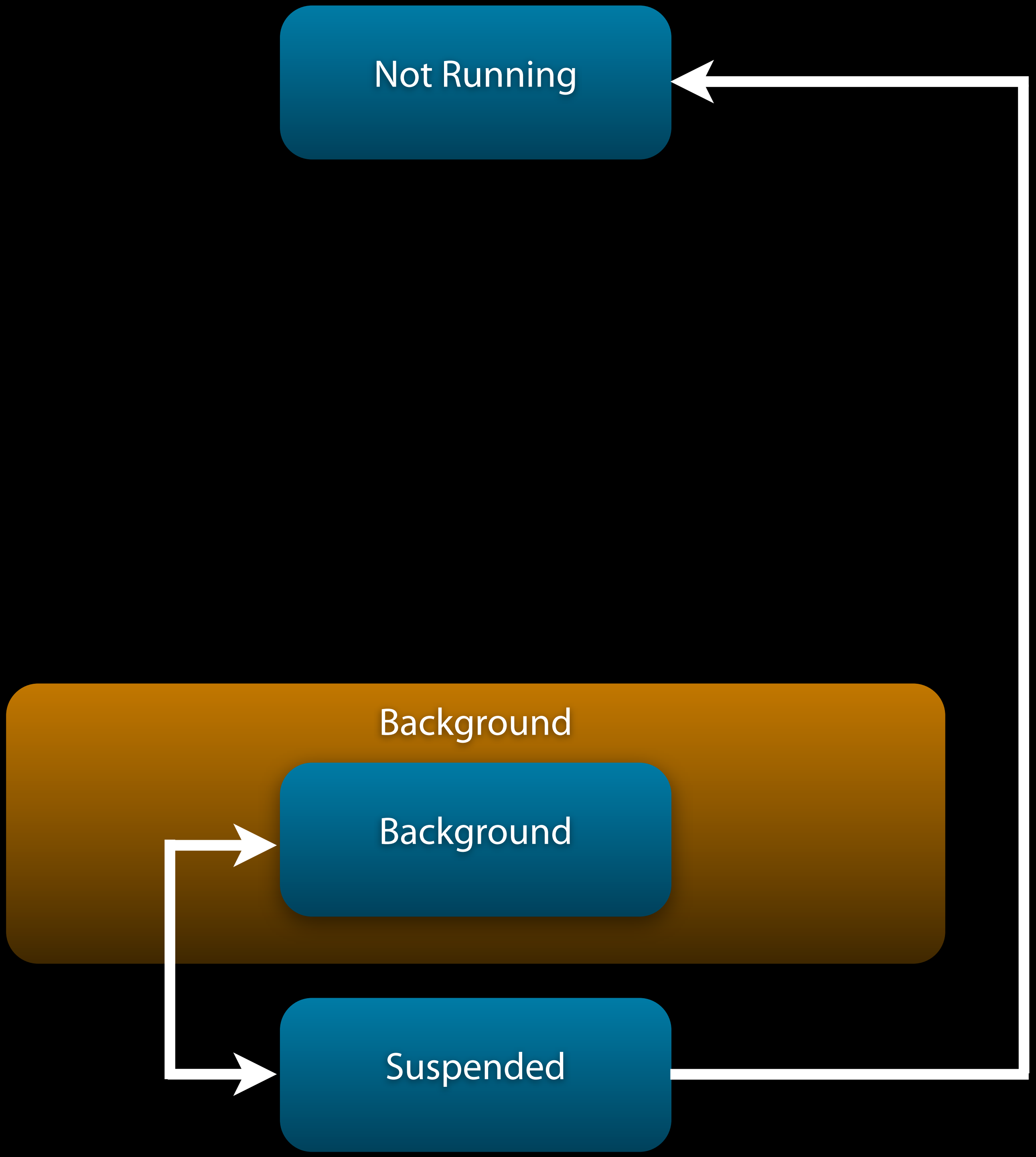


Multitasking

Long running background tasks

- Playing audio content
- Location tracking
- VoIP
- Receive regular updates from external accessories
- Finite length tasks

Multitasking
Long running background tasks



Application Life Cycle

Events that affect

- Incoming call
- Alert dialogs
    - Text Messages
    - Push Notifications
    - Calendar Reminders

UIApplication
Overview

- Represents the iOS Application in the OS
- UIResponder class for everything application level specifics

Views

What we see is not necessarily what we get

Views
Overview

- Layout and subview management
- Drawing and animation
- Event handling
    - Touch events
    - Responder Chain

UIWindow
The root view

- Special UIView subclass
- Contains Application's content
- Works with view controllers to facilitate orientation changes
- Spans entire main screen
- Can create more to show content on external displays

UIView

Concept

```
CGRect frame = CGRectMake(xpos, ypos, width, height);
UIView *newView = [[UIView alloc] initWithFrame:frame];

newView.layer.cornerRadius = 10.0;
```

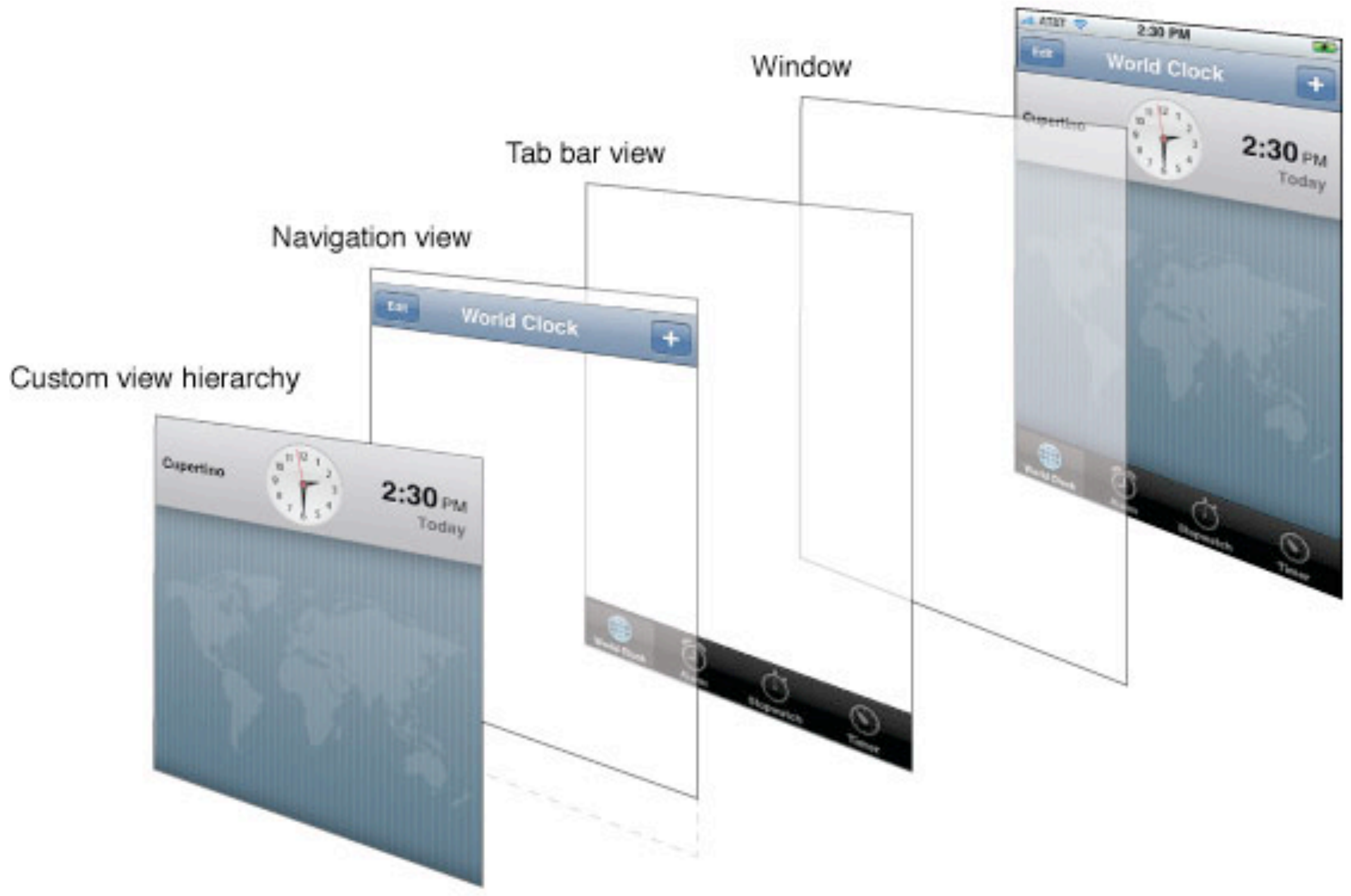
UIView

Concept

- Data model for display
- Values set on the UIView synched with internal CALayer
- CALayer represent buffered drawing in video memory
  - OS manipulates CALayer
  - Basis of Core Animation
- Manages subviews

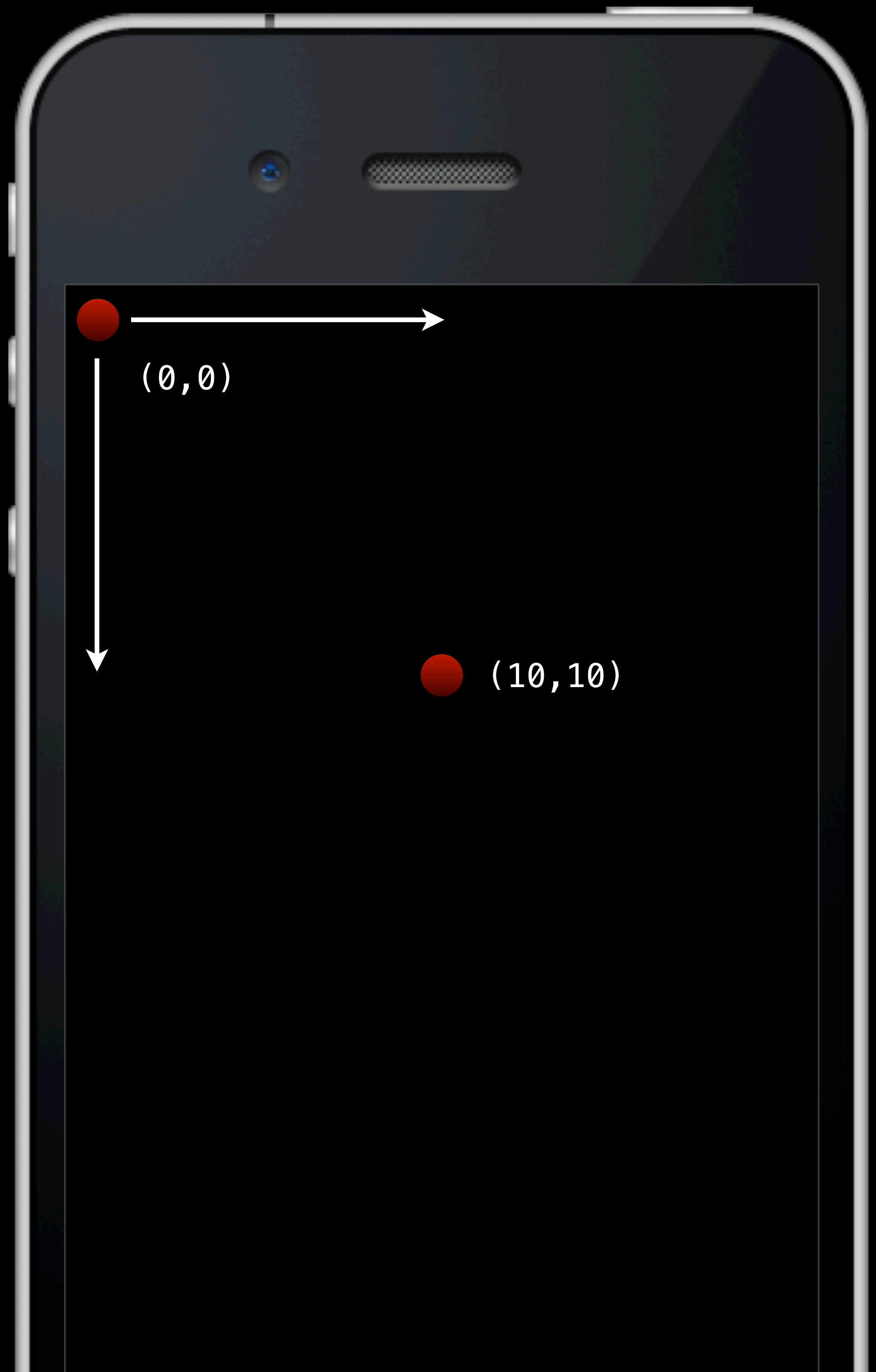
UIView

Layers



UIView

Coordinate System



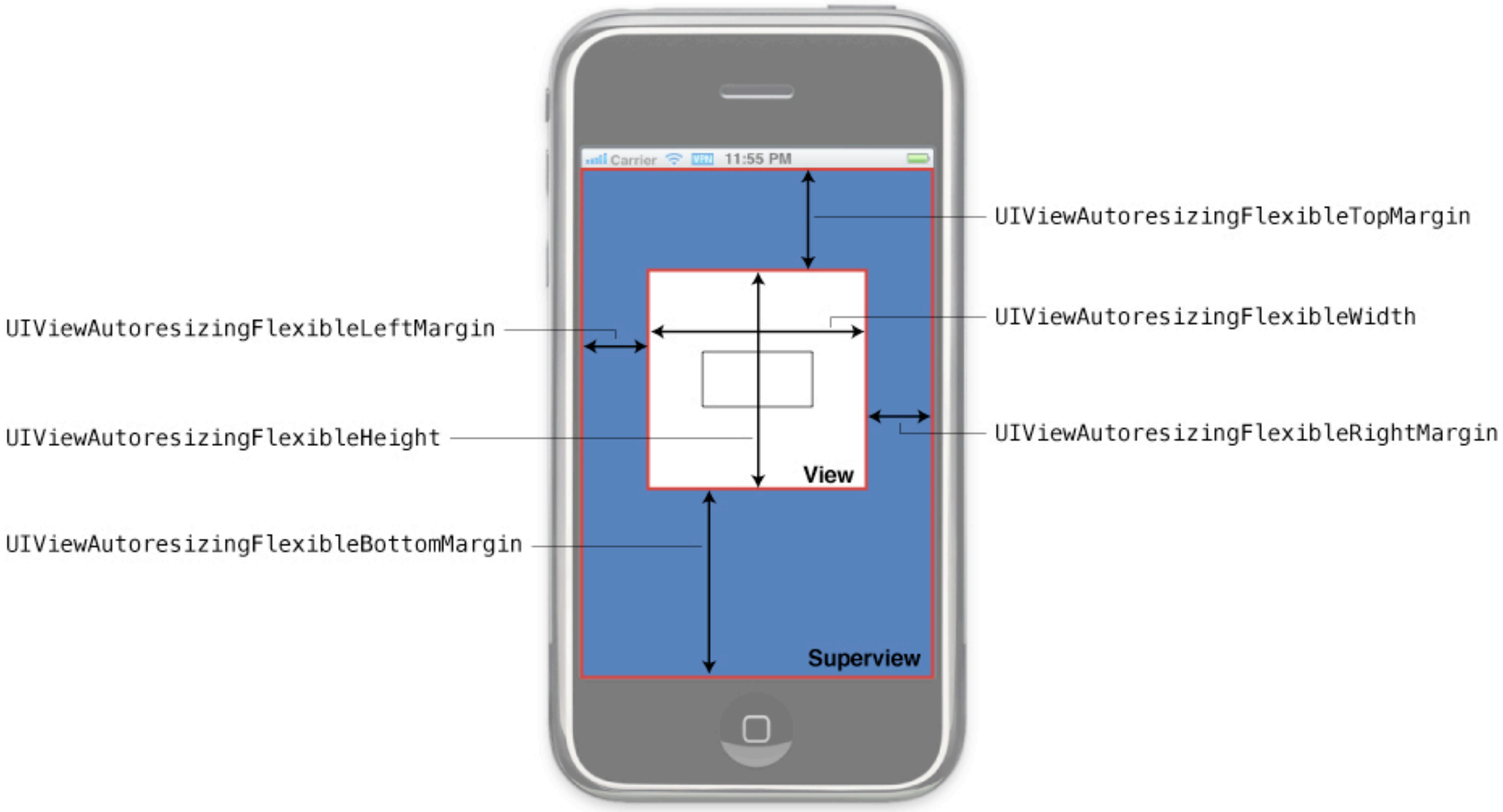
UIView
Coordinate System

- Points on a view can be converted for another view
- Converts from target view to local coordinate of current
- Scale and rotate CALayer representation of UIView

UIView
Size and position

- Can be modified at runtime by changing `frame` attribute
- Automatic resizing in response to superview can be set
- Modes for dealing with when a view is resized

UIView
Size and position



Carrier
11:55 PM
UIViewAutoresizingFlexibleTopMargin
UIViewAutoresizingFlexibleWidth
UIViewAutoresizingFlexibleLeftMargin
UIViewAutoresizingFlexibleRightMargin
UIViewAutoresizingFlexibleHeight
View
UIViewAutoresizingFlexibleBottomMargin
Superview

UIView
Drawing and Layout

- Required to subclass UIView
- Drawing deals with rendering
    - Core Graphics draw calls
    - A way to “flatten” layers
- Layout deals with size and position of subview
- Force refresh by flagging as dirty

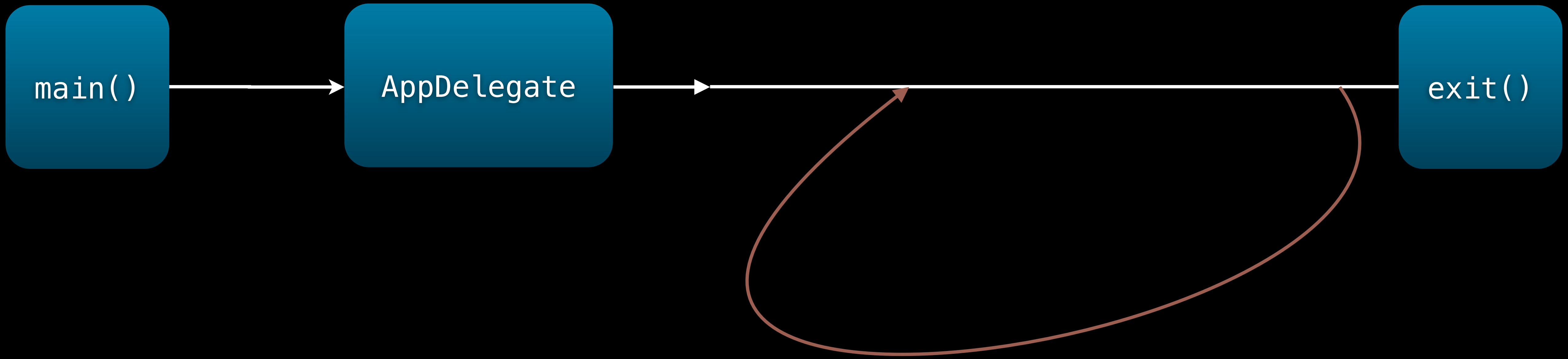
UIView

Other properties

- Visibility
    - alpha
    - opaque
    - hidden
- Background Color

Events

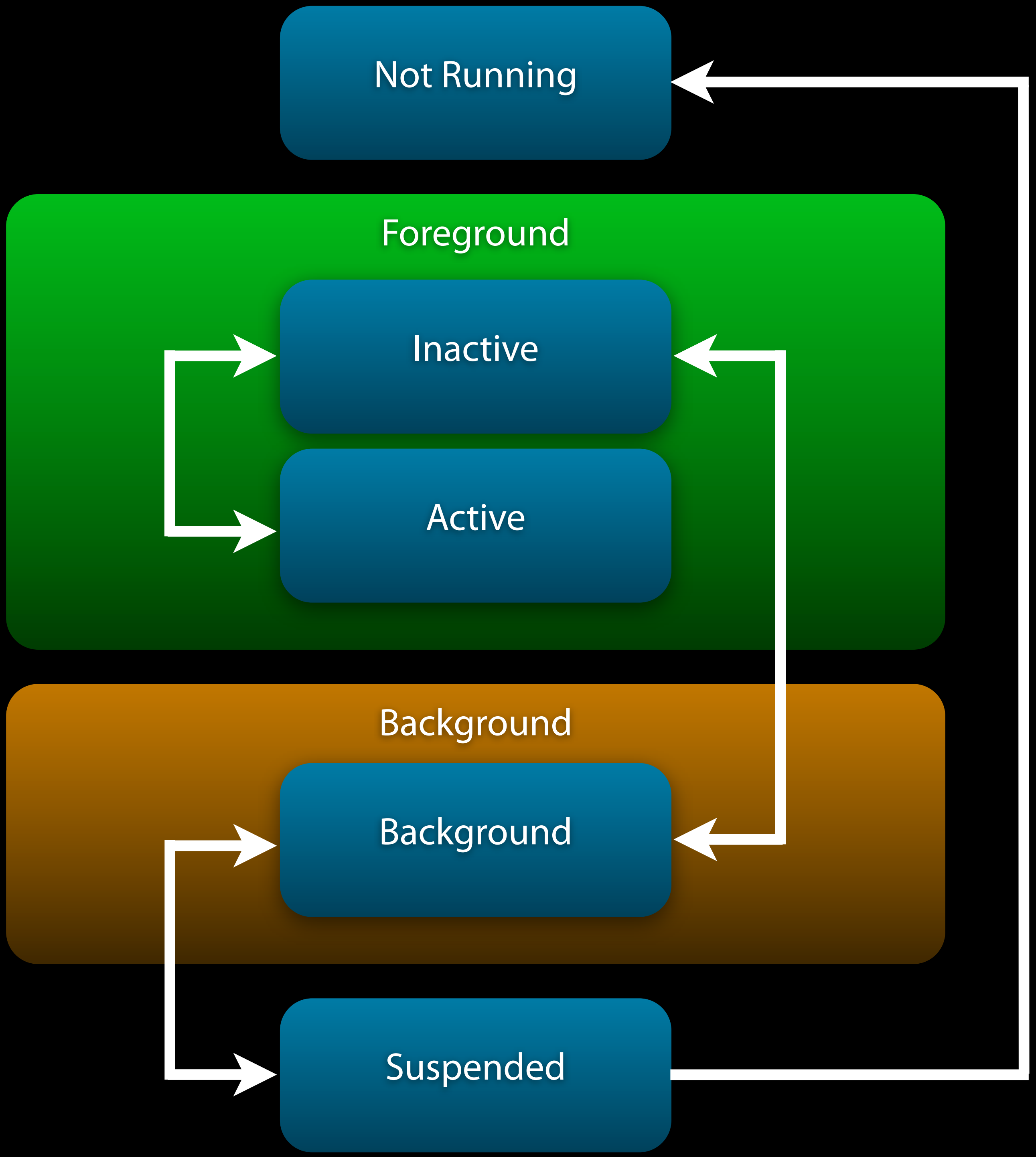
What we see is not necessarily what we get



NSRunLoop

Multitasking

Events


Not Running
Foreground
Inactive
Active
Background
Background
Suspended

Events
Overview

- Mechanisms for handling events
- Application Events
- Touch Events
- Local/Remote Notifications

Event Mechanisms

Delegate

- 1-to-1 relationship between delegate and sender
- Low latency
- Usually called on same thread

Event Mechanisms

Delegate

- Uses delegate pattern to receive messages
- Delegate usually stored as a weak ivar
- Good to check if delegate responds to a message

```
if ([delegate respondsToSelector:@selector(delegateSelector:)]) {
    [delegate delegateSelector:message];

}
```

Event Mechanisms

NSNotification

- 1-to-many triggering
- Higher latency
- Non deterministic thread

Event Mechanisms

NSNotification

- NSNotificationCenter is a post office for messages
- Post NSNotification
- Unsubscribe/Subscribe to NSNotificationCenter

```
NSNotification *notification = [NSNotification notificationWithName:<#(NSString *)#>
                                                             object:<#(id)#>
                                                           userInfo:<#(NSDictionary *)#>];

[[NSNotificationCenter defaultCenter] postNotification:<#(NSNotification *)#>];
```

Application Events

OS → Application Messages

- Multitasking states
- System Notifications
    - Low memory warnings
    - Significant Time Change

Application Events

OS → Application Messages

- Multitasking states
- System Notifications
- Opening URL resource
    - Inter process messaging scheme
    - Set protocol to match in `info.plist`
    - Target app will launch when URL is opened

Application Events

OS → Application Messages

- Multitasking states
- System Notifications
- Opening URL resource
- Status bar changes
- Protected Content Changes

Application Events

OS → Application Messages

- Multitasking states
- System Notifications
- Opening URL resource
- Status bar changes
- Protected Content Changes
  - Files can be marked to be encrypted or “protected”
  - Different events to decrypt it

Touch Events

Finger power

- Touch events delivered by UIWindow to UIViews
- UIViews can turn off user interaction
- UIViews can have gesture recognizers attached
- Can pass along action via responder chain

Touch Events

Finger power

- Subclass to deal with touch events
    - `touchesBegan:withEvent:`
    - `touchesMoved:withEvent:`
    - `touchesEnded:withEvent:`
    - `touchesCancelled:withEvent:`

Gesture Recognizers

Next level touch events

- Predefined gestures that can be recognized
- Set parameters that satisfy the gesture
- Set a target and action
- Add to UIView

Notifications

Remote Push Notifications

- Event generated when Push notification is received
- 2 receive states
  - App is active
  - App is inactive/background/not started

Remote Notifications

When active

- Payload in userinfo dictionary
- Event passed to `application:didReceiveRemoteNotification:`
- Always when `application:didFinishLaunchingWithOptions:` not implemented

Remote Notifications
When inactive

- System alert notification popup
- Action button launches app with payload
- Event handled by `application:didFinishLaunchingWithOptions:`

Local Notifications

No server involved

- Notifications are scheduled
- UILocalNotification as payload
- No need to register
- Launch behavior is similar to Remote Notifications

UIKit/Foundation

The resources you will be using

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
        - `NSString, NSAttributedStrings, NSMutableString`

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
    - Collections
        - `NSArray, NSSet, NSDictionary`

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
    - Collections
    - File access
        - `NSFileManager`

Foundation
Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
    - Collections
    - File access
    - Networking
        - `NSURLConnection`

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
    - Collections
    - File access
    - Networking
    - Deserialization/Serialization
        - `NSCoding, NSKeyedArchiver, NSKeyedUnarchiver`

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
    - UTF8 Strings
    - Collections
    - File access
    - Networking
    - Deserialization/Serialization
    - Date/Time
        - `NSDate, NSCalendar`

Foundation

Base Objective-C Framework

- Foundation provides base cross platform implementation
- A helpful framework
  - UTF8 Strings
  - Collections
  - File access
  - Networking
  - Deserialization/Serialization
  - Date/Time
  - Introspection
    - `@selector, NSInvocation, NSClassFromString`

UIKit

iOS Framework

- UIKit provides iOS specifics by building on Foundation
    - Mainly User Interface classes
    - Application specific implementation

Others

System Frameworks

- Core *
    - Core Animation
    - Core Location
    - Core Graphics
- MapKit
- AddressBook
- EventKit
- WebKit

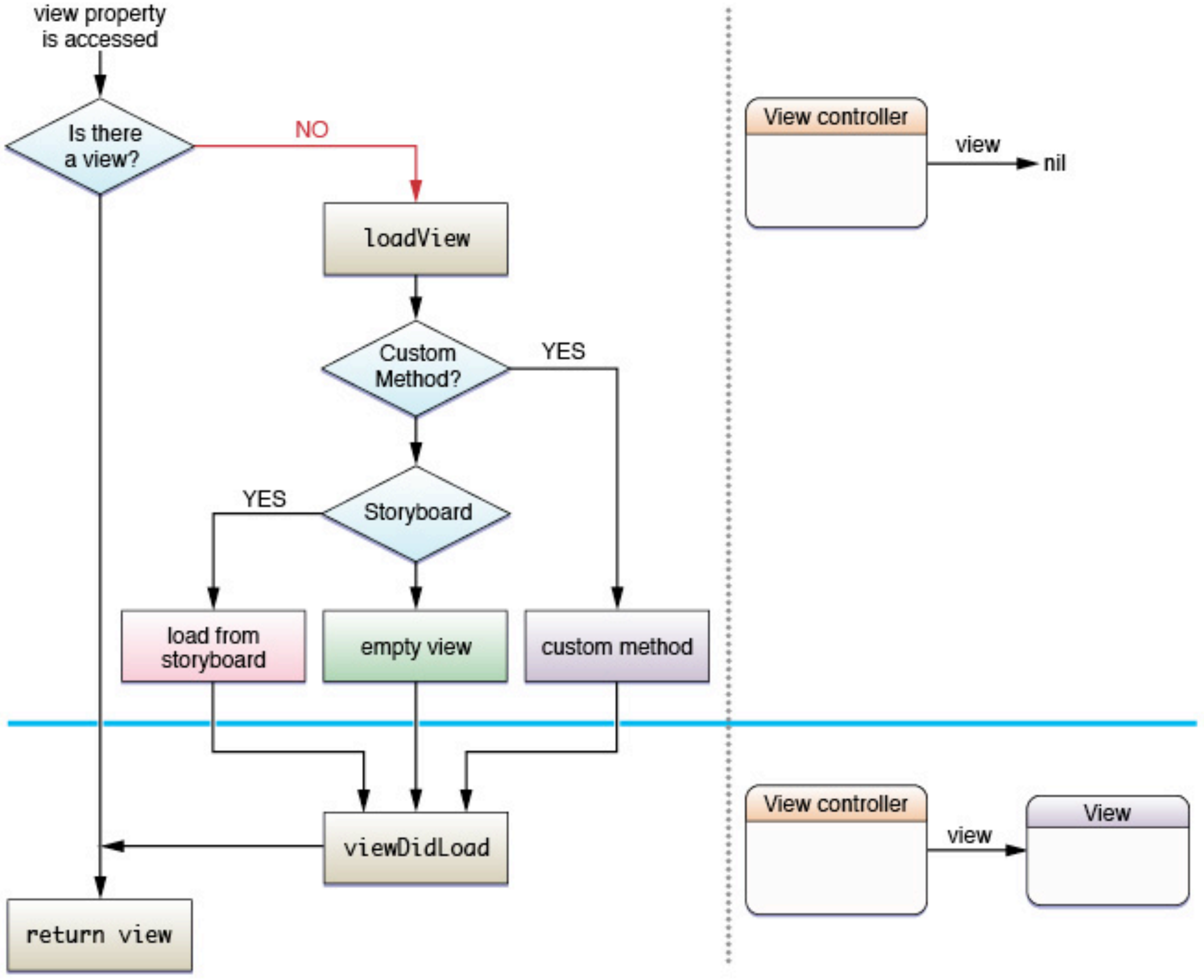
UIViewController

The C of the MVC

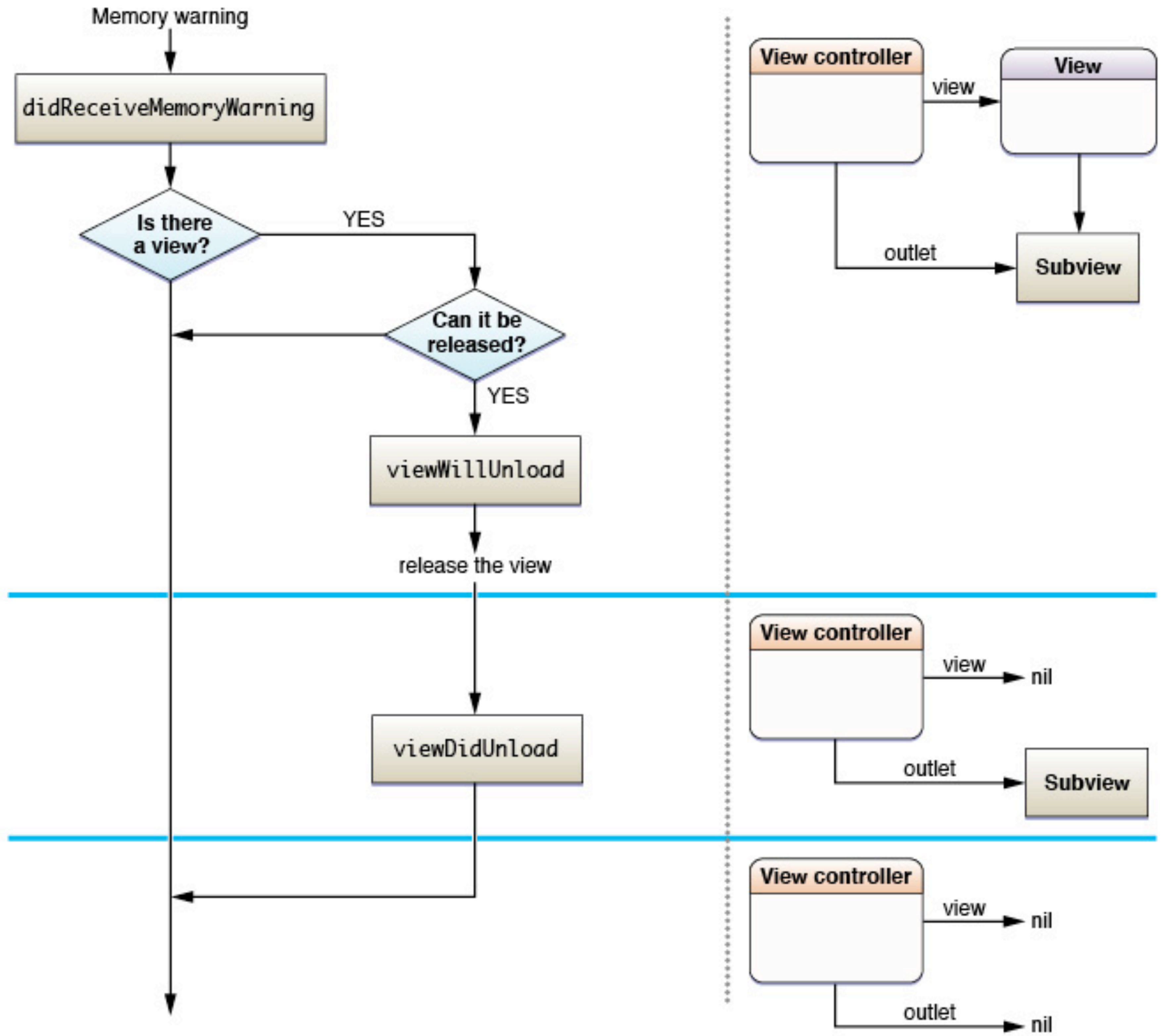
- Base class for most view controllers
- Provides event handling for views
    - Loading

UIViewController

Loading



UIViewController
Unloading



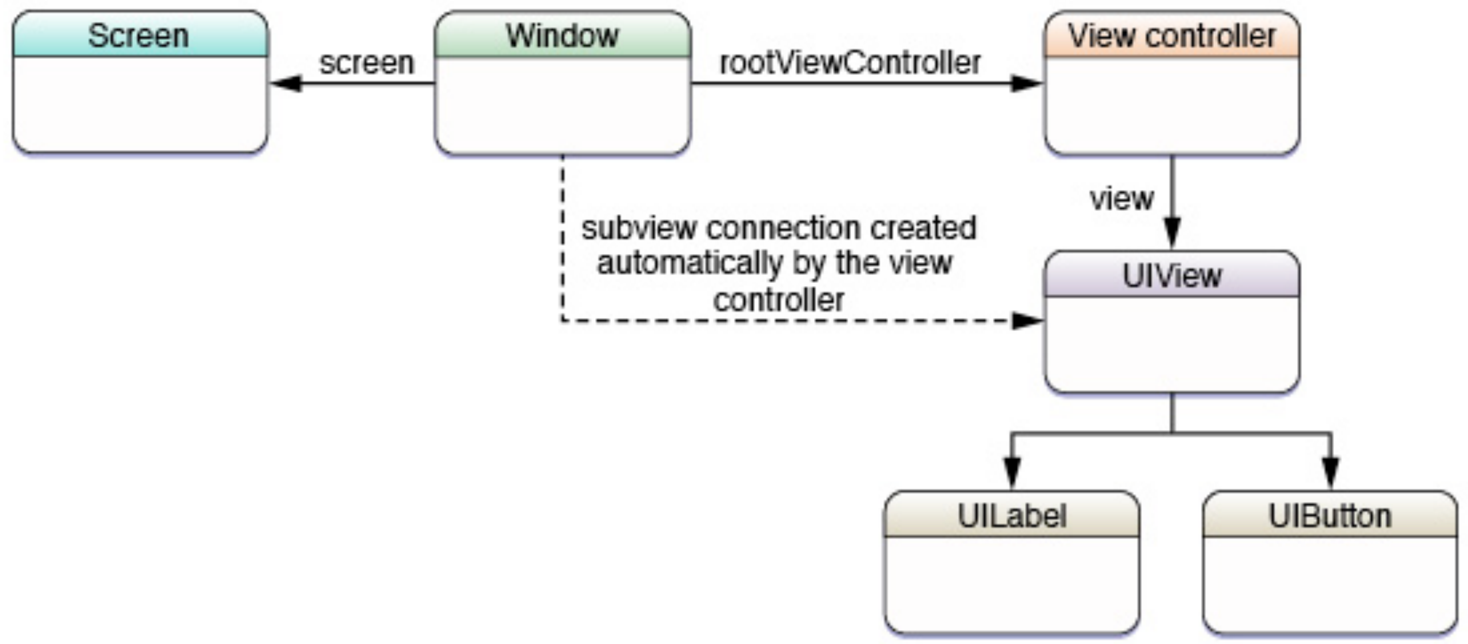
UIViewController

The C of the MVC

- Base class for most view controllers
- Provides event handling for views
    - Loading
    - Interface orientation changes
    - Manages other view controllers

UIViewController

View controllers manage views



UIViewController

The C of the MVC

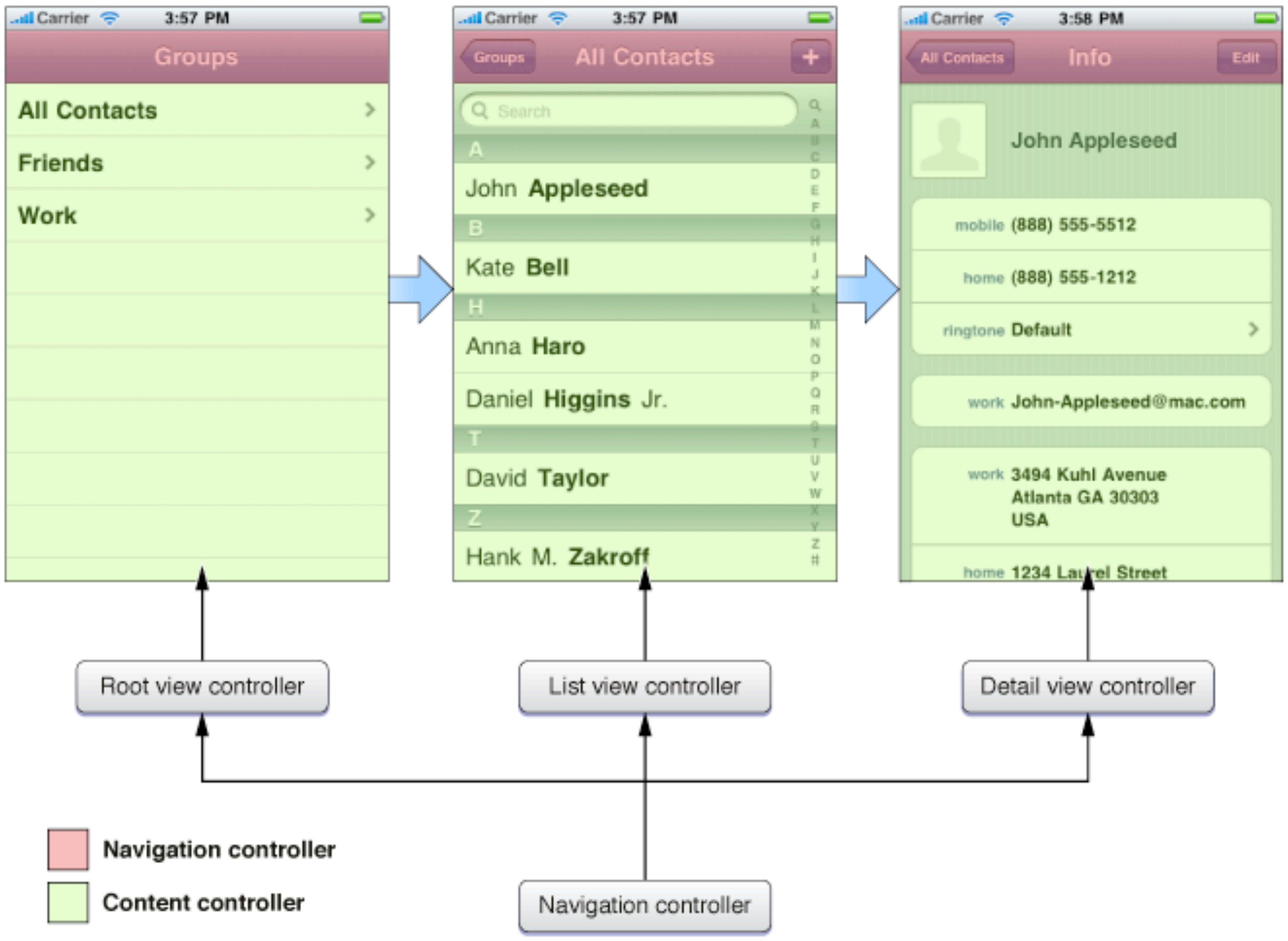
- Base class for most view controllers
- Provides event handling for views
    - Loading
    - Interface orientation changes
    - Manages other view controllers
    - Presenting view controllers
    - Low Memory Warnings

UINavigationController
Workflow control

- Provides a system to structure view controllers hierarchically

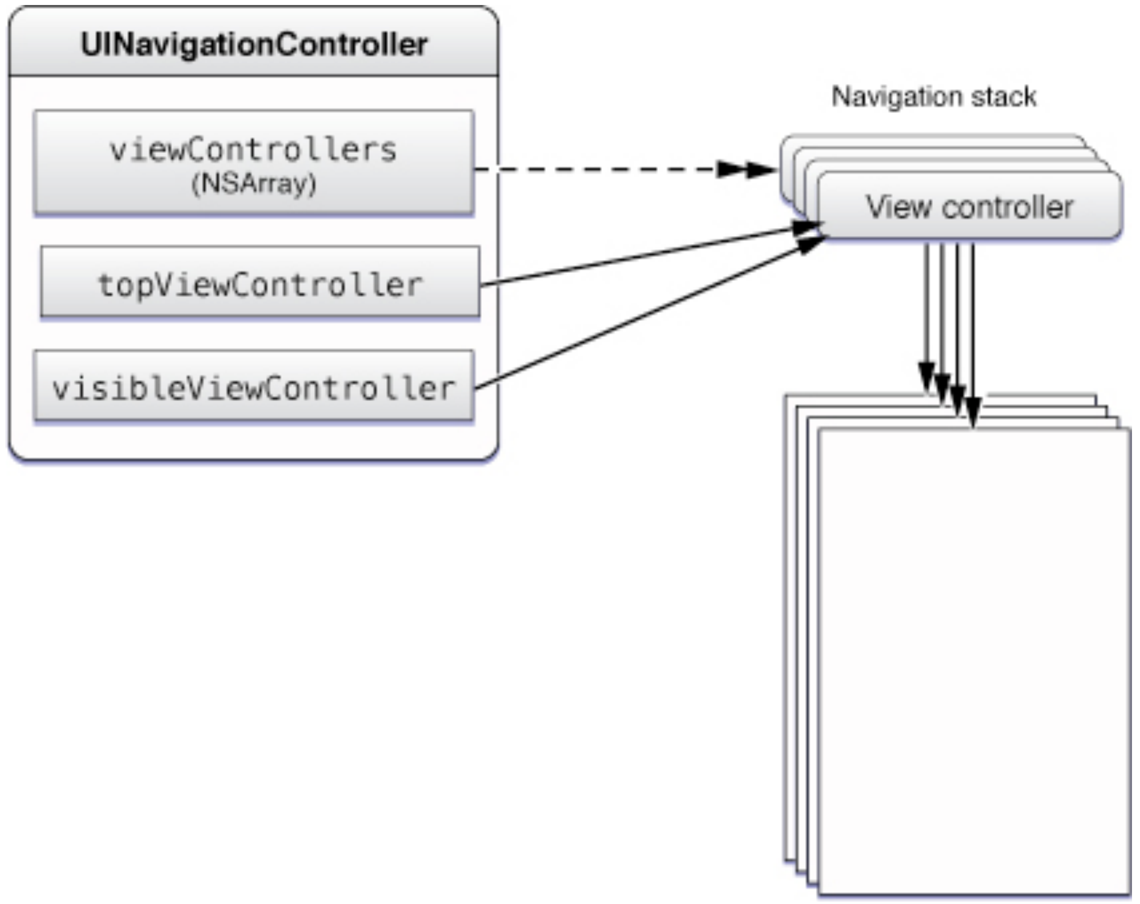
UINavigationController

Organizes View Controllers



UINavigationController

Navigation Stack



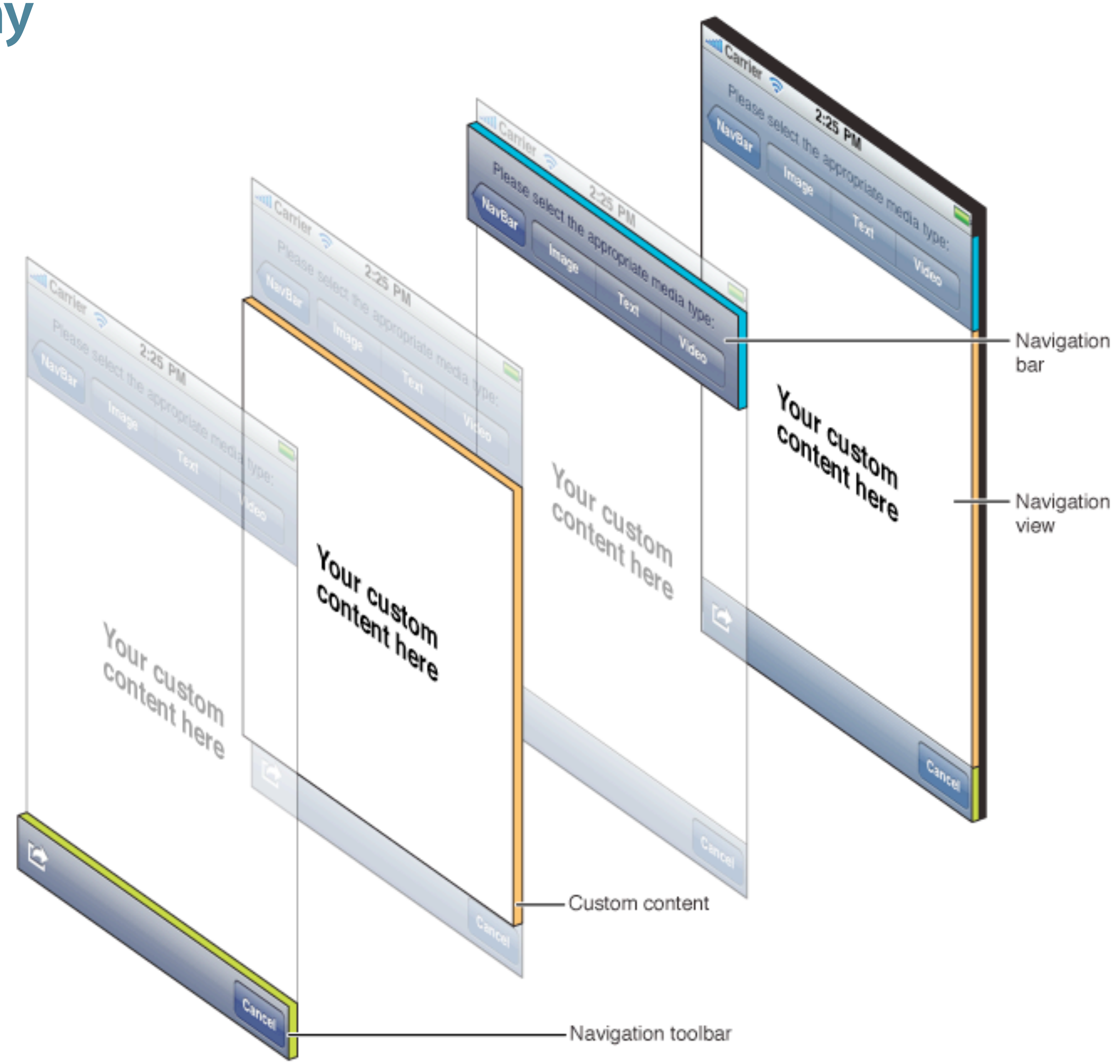
UINavigationController

Workflow control

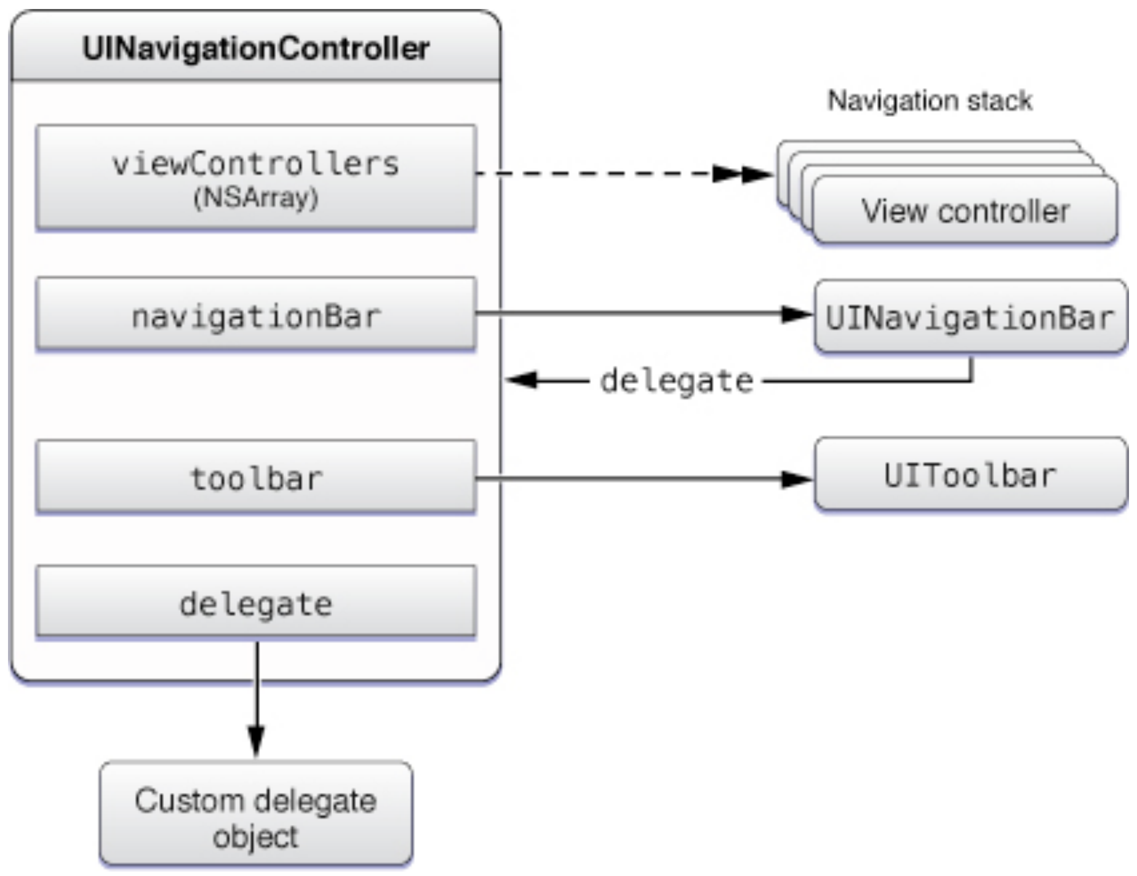
- Provides a system to structure view controllers hierarchically
- Provides UI elements that view controllers can configure

UINavigationController

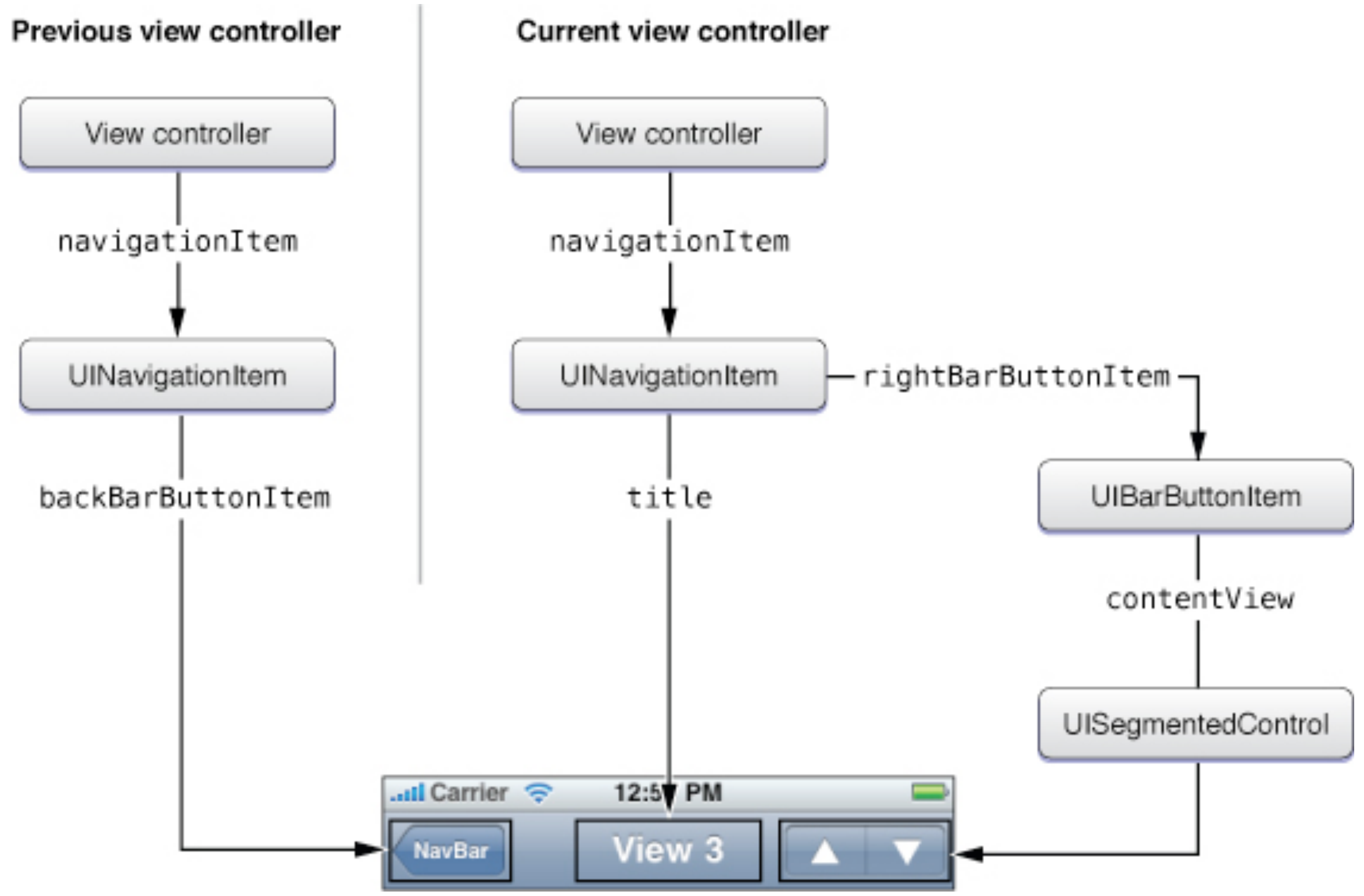
Anatomy



UINavigationController
Composition

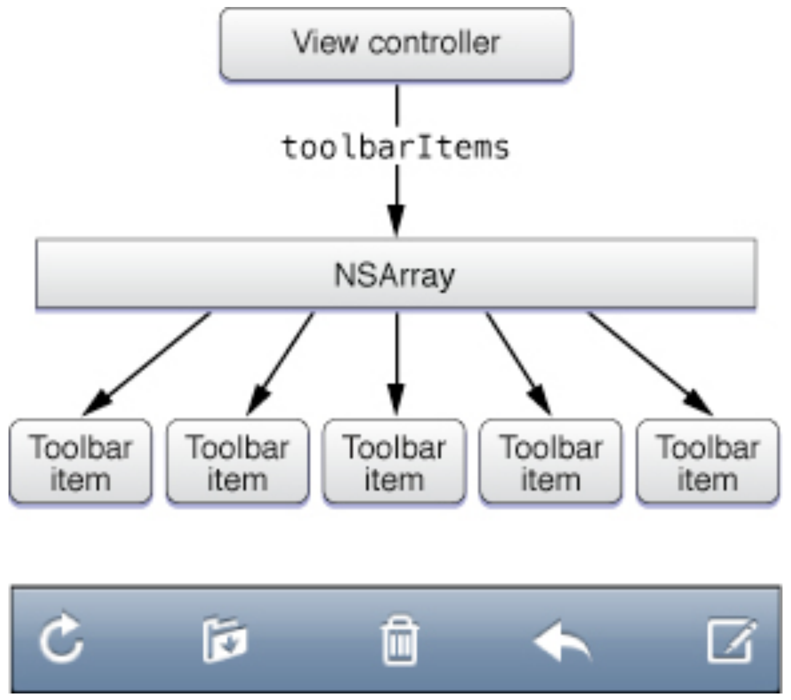


UINavigationBar
Anatomy



UIToolBar

Anatomy



UINavigationController

Workflow control

- Provides a system to structure view controllers hierarchically
- Provides UI elements that view controllers can configure
    - Settable via View Controller properties
    - Or manually added to a view

UINavigationController
Workflow control

- Provides a system to structure view controllers hierarchically
- Provides UI elements that view controllers can configure
- Uses animations to show and hide workflow

```
[navigationController pushViewController:newViewController animated:YES];
```

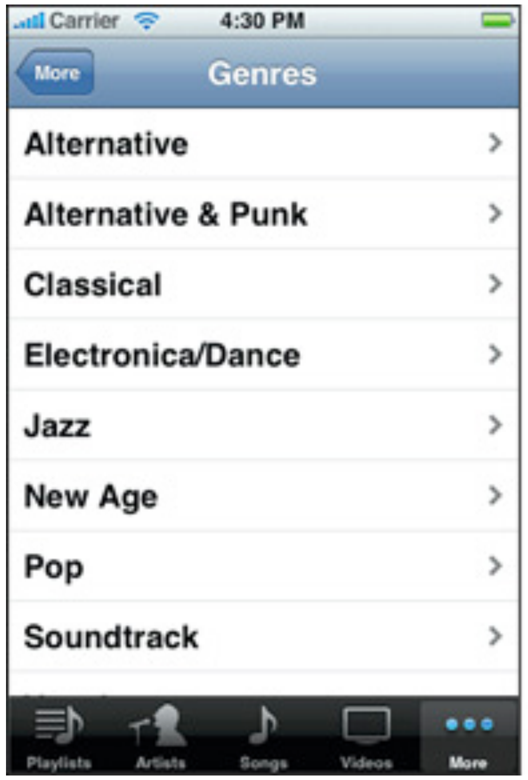
UITableView

The table manager

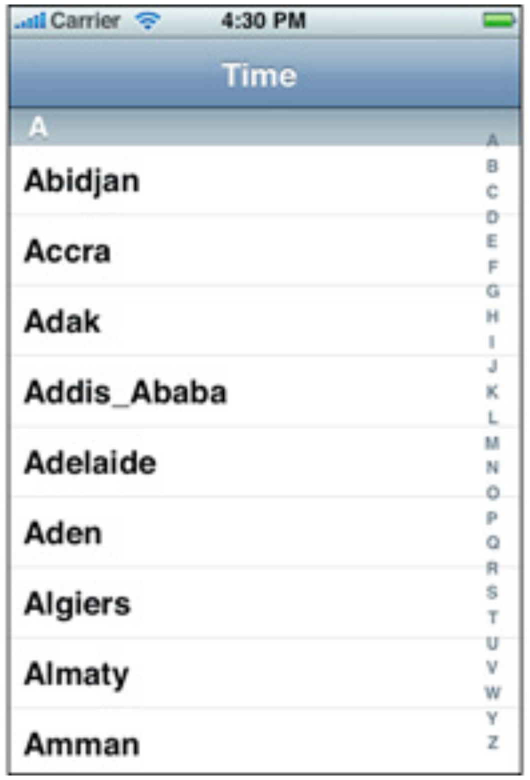
- UIScrollView subclass
- Manages a group of views (cells)
- Uses delegate pattern to update and configure cells
- High performance
- Most de-facto way to present information

UITableView

Various styles

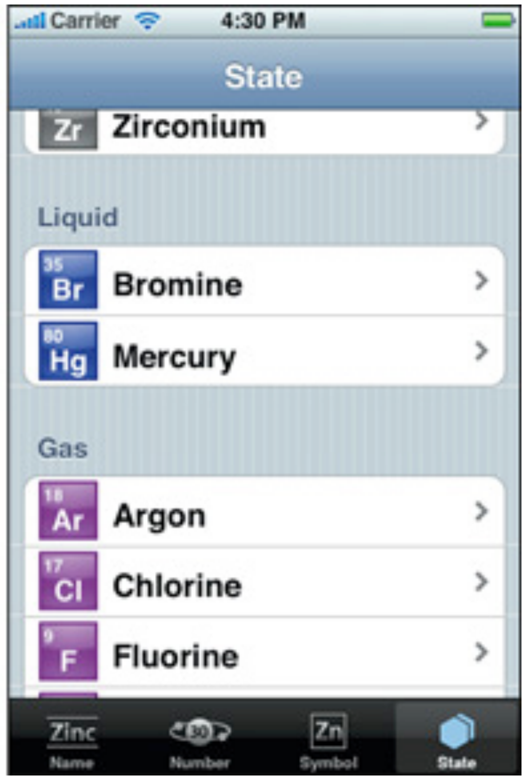


Alternative
Alternative & Punk
Classical
Electronica/Dance
Jazz
New Age
Pop
Soundtrack



A

Abidjan
Accra
Adak
Addis_Ababa
Adelaide
Aden
Algiers
Almaty
Amman



Zr Zirconium

Liquid

35 Br Bromine
80 Hg Mercury

Gas

18 Ar Argon
17 Cl Chlorine
9 F Fluorine

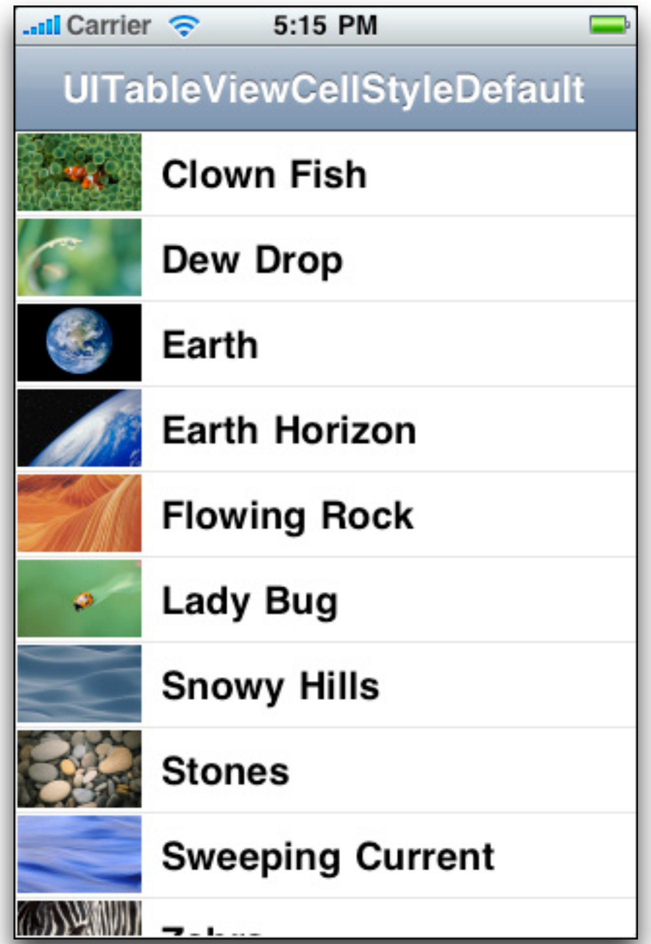
UITableView

Sections and Rows

- Contains many sections
- Each section contains many rows
- Each section also has a header and a footer view
- Each row is 1 UITableViewCell

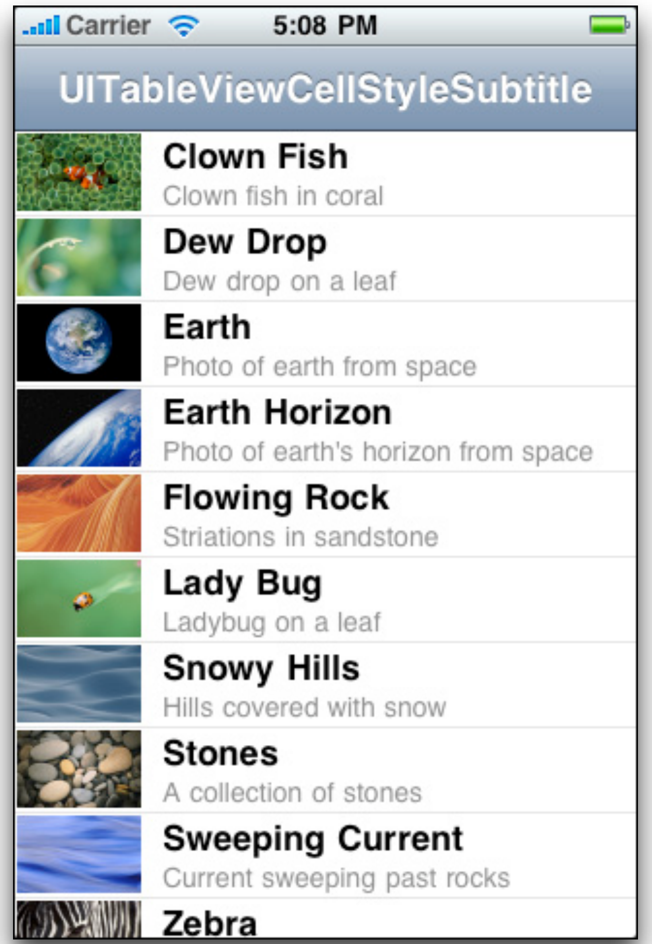
UITableViewCell

Various styles


Carrier 5:15 PM

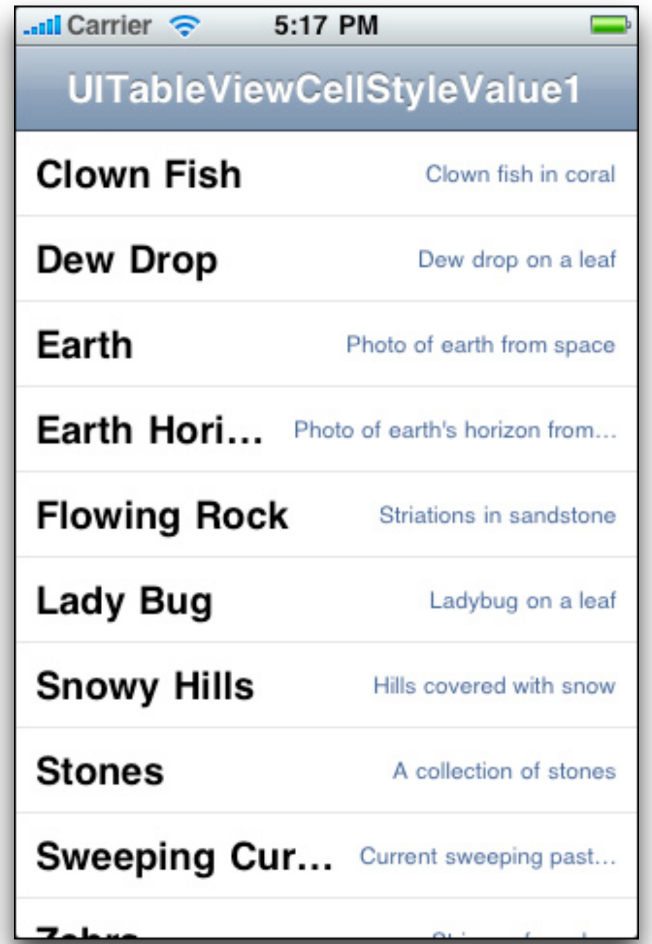
UITableViewCellStyleDefault

Clown Fish
Dew Drop
Earth
Earth Horizon
Flowing Rock
Lady Bug
Snowy Hills
Stones
Sweeping Current


Carrier 5:08 PM

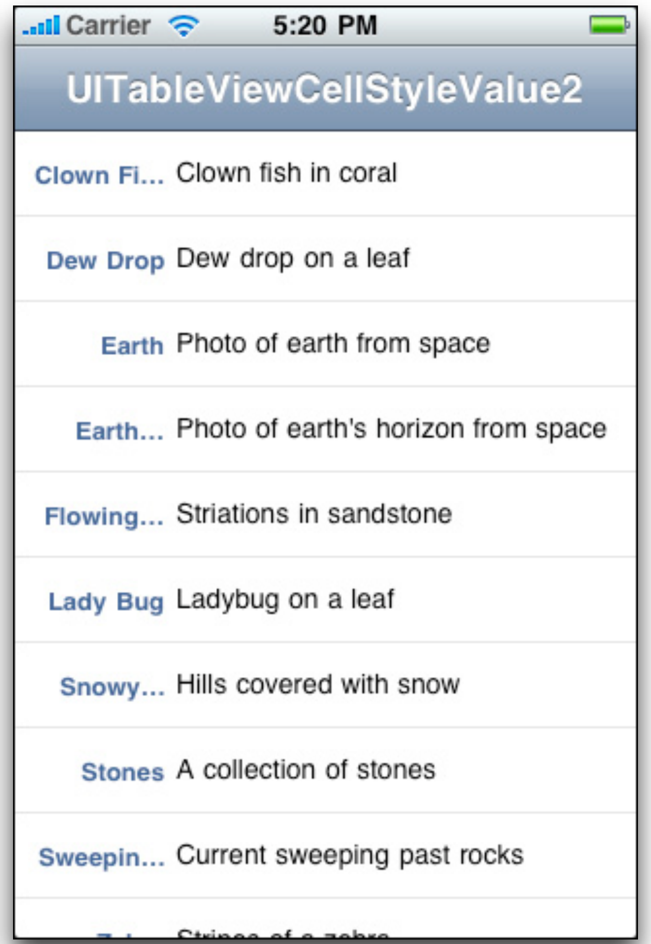
UITableViewCellStyleSubtitle

Clown Fish	Clown fish in coral
Dew Drop	Dew drop on a leaf
Earth	Photo of earth from space
Earth Horizon	Photo of earth's horizon from space
Flowing Rock	Striations in sandstone
Lady Bug	Ladybug on a leaf
Snowy Hills	Hills covered with snow
Stones	A collection of stones
Sweeping Current	Current sweeping past rocks
Zebra	


Carrier 5:17 PM

UITableViewCellStyleValue1

Clown Fish	Clown fish in coral
Dew Drop	Dew drop on a leaf
Earth	Photo of earth from space
Earth Hori...	Photo of earth's horizon from...
Flowing Rock	Striations in sandstone
Lady Bug	Ladybug on a leaf
Snowy Hills	Hills covered with snow
Stones	A collection of stones
Sweeping Cur...	Current sweeping past...


Carrier 5:20 PM

UITableViewCellStyleValue2

Clown Fi...	Clown fish in coral
Dew Drop	Dew drop on a leaf
Earth	Photo of earth from space
Earth...	Photo of earth's horizon from space
Flowing...	Striations in sandstone
Lady Bug	Ladybug on a leaf
Snowy...	Hills covered with snow
Stones	A collection of stones
Sweepin...	Current sweeping past rocks

Various Accessory View

› ● ✓

UITableViewController

The table manager

- Implements the delegate and data source for table views
- Manages a UITableView instead of a generic view
- Can be added to a navigation hierarchy easily

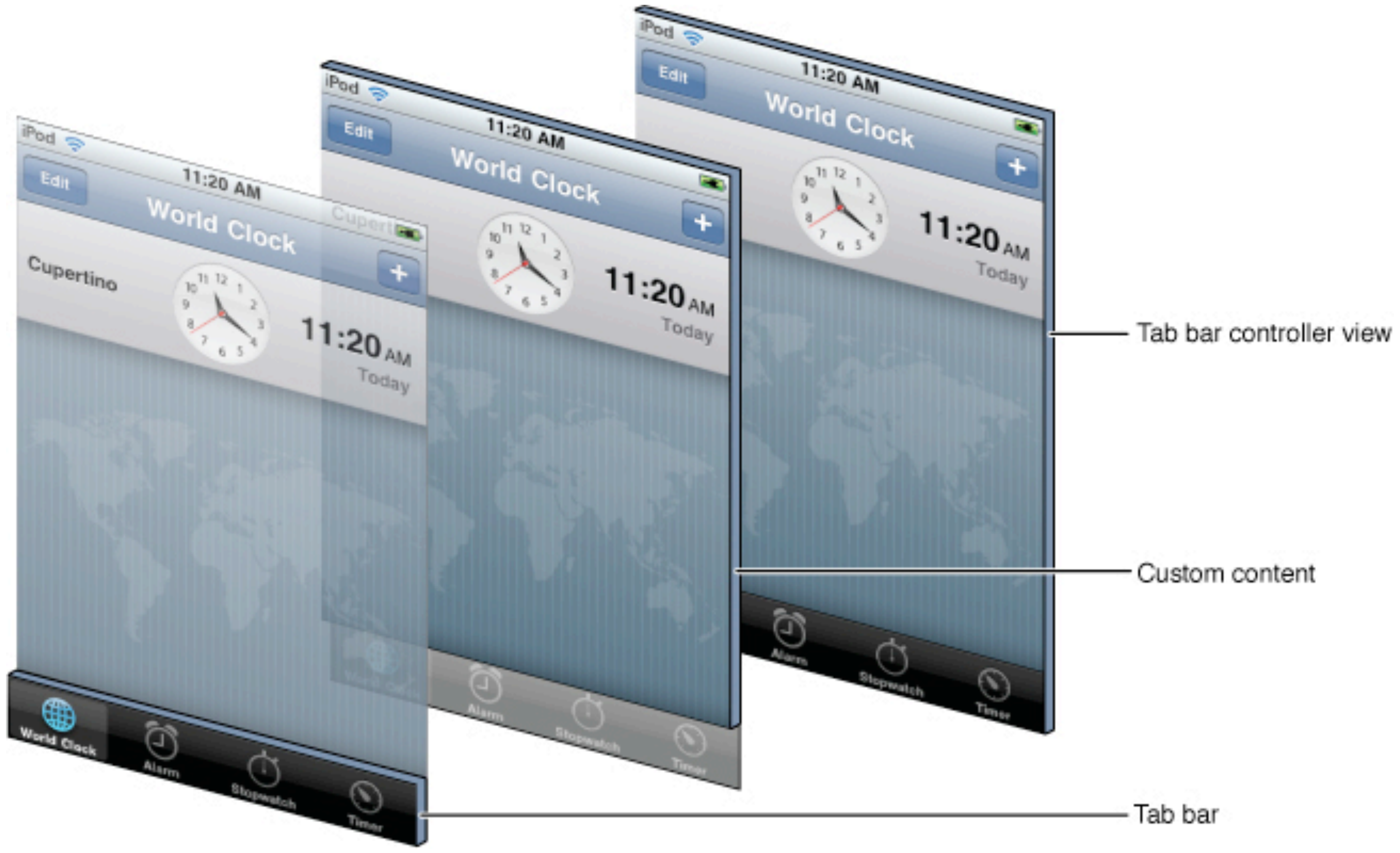
UITabBarController

Sectional organization

- Manages set of views that can be toggled around
- Concept of “sections”

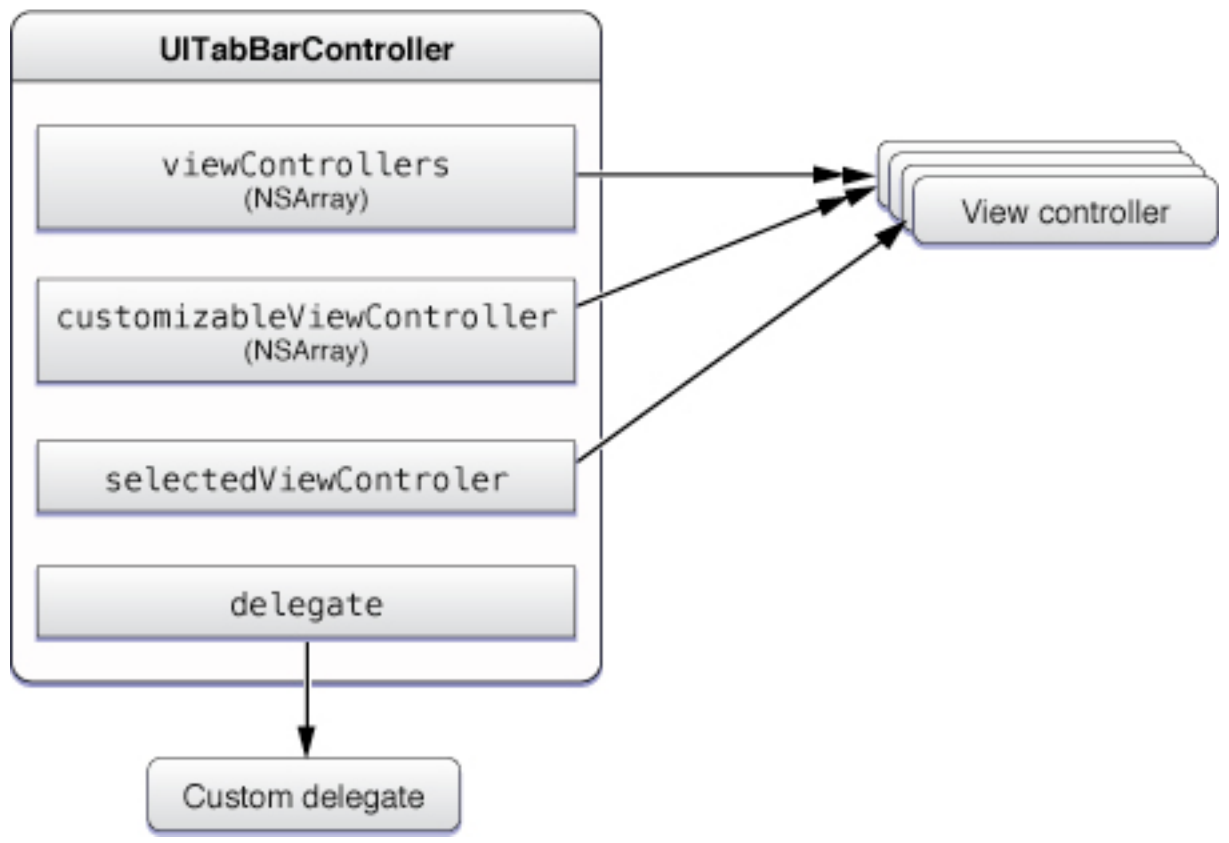
UITabBarController

Anatomy



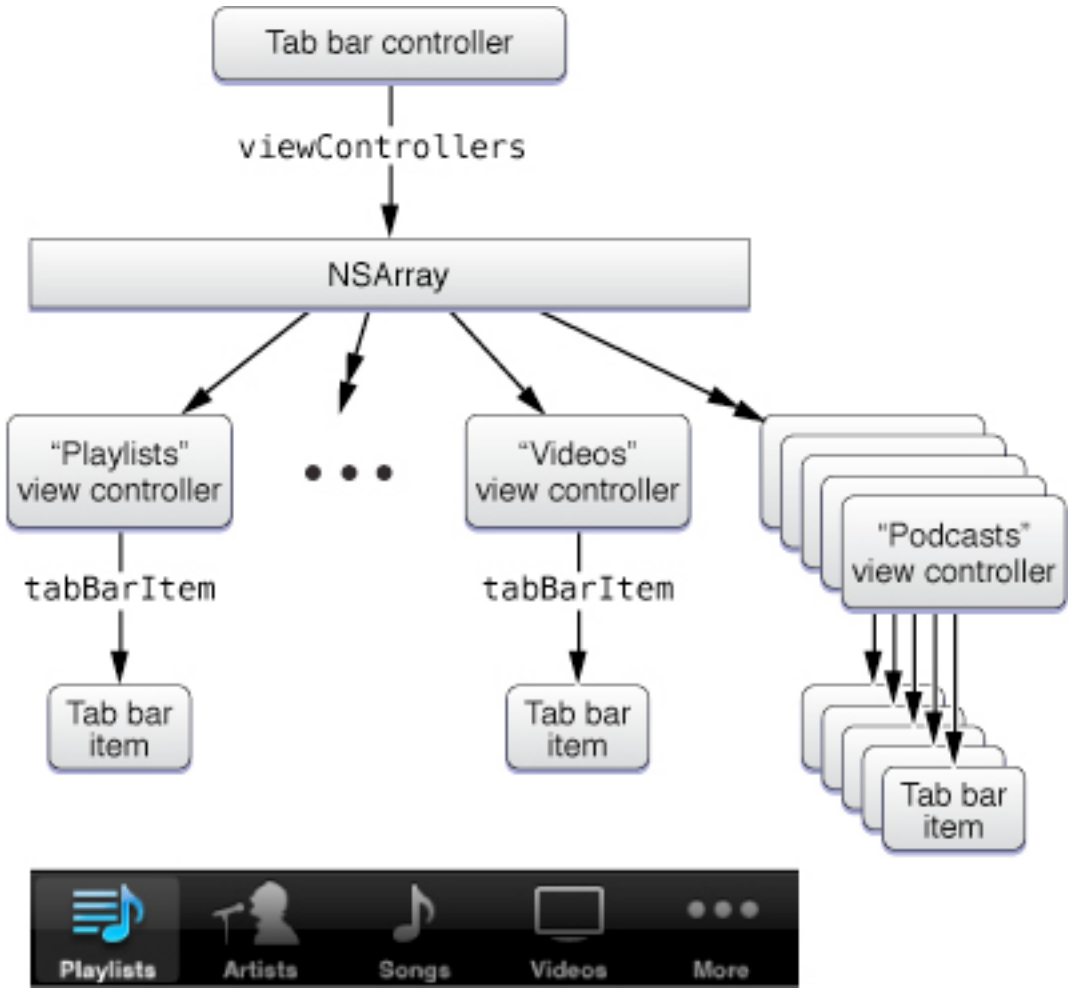
UITabBarController

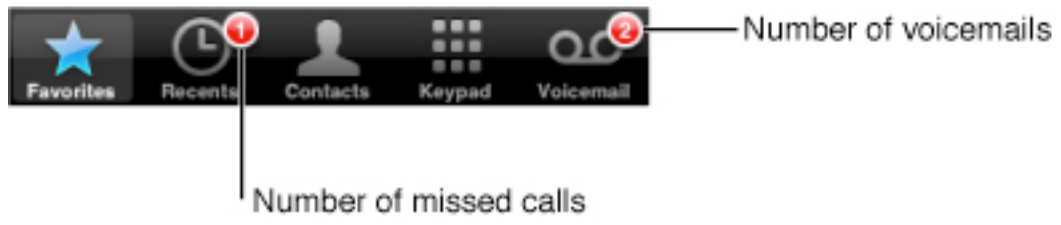
Composition



UITabBarController

Items





Performance

If its not fast, its not good

Performance

Why is it important

- Slow is a bad user experience
- Native code = you are more in control
- Awesome instrumentation tools
    - Drawing performance
    - Memory usage
    - Leaks/Zombies
    - Network use

Performance

Memory

- Creating a new object is more expensive
- Clear out old values and reuse objects
- UITableView is a great example
- Less leaks = better performance

Performance

Flattening Layers

- Compositing is awesome but expensive
- Render non interactive elements in one pass
- `-drawRect`
- Reduce usage of transparent views

Performance

Images

- Always use `[UIImage imageNamed:@”image”]`
- Internal caching mechanisms
- Autoloading of @2x images

Performance

Animation

- Provides the illusion of performance
- Strict adherence to animation duration
- Animate while continuing processing/loading

Performance

Don’t touch the main thread

- Main thread is for UI and is watchdog’ed
- Move stuff onto secondary threads
    - Dispatch Queues
    - NSOperation

Performance

Floating point math

- Vector instruction set in ARM CPUs
- Accessible via Accelerate framework

Performance

Tuning your code

- Test on device(s)
- Simulator is orders of magnitude faster

Performance

Reduce power consumption

- Don’t poll, use events and call backs
- Batch network requests together when on 3G
- Turn off sensors when not in use