Application Life Cycle

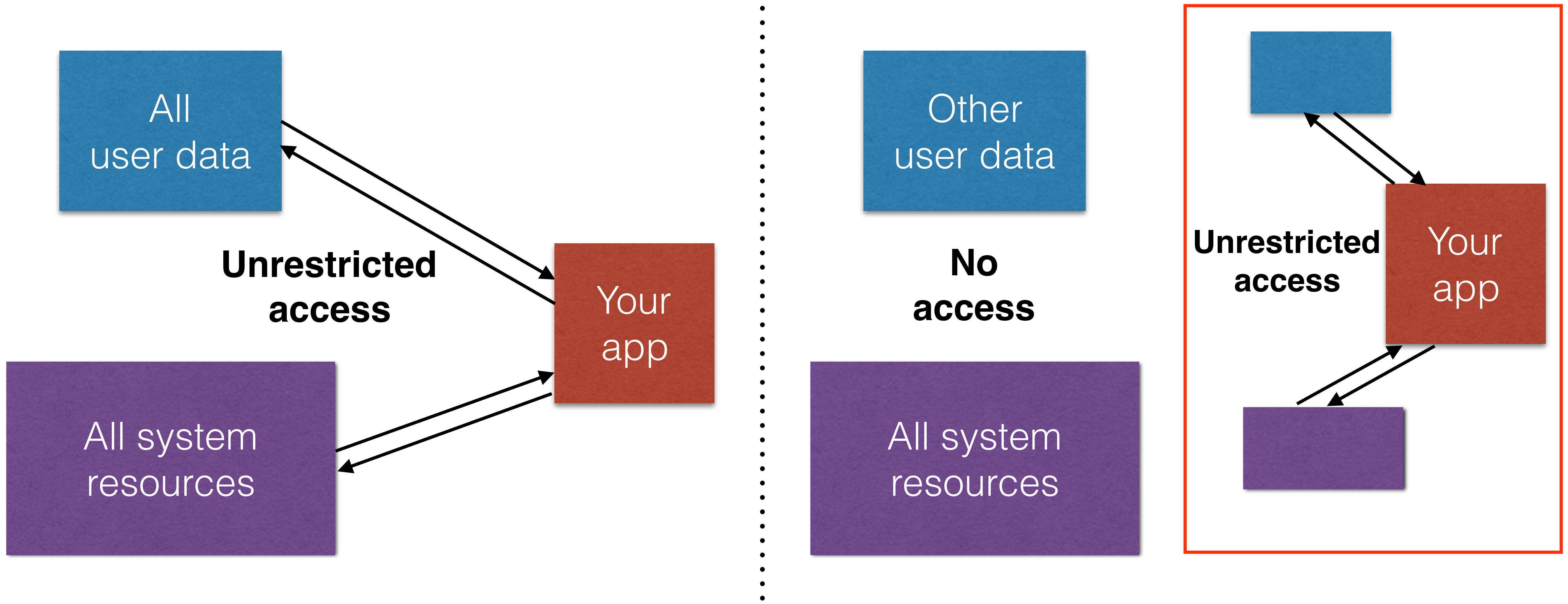
Lecture 4
by Michael Kramskoy

What we will go through in this lecture

- **Sandbox**
- **MVC**
- **UIApplication**
- **AppDelegate**
- **Application states**
- **We will create a simple iOS application**
- **We will run it in a simulator**

Sandbox

Definition from Wikipedia: **sandbox** is a security mechanism for separating running programs… may be seen as a specific example of virtualisation.



Architecture Patterns

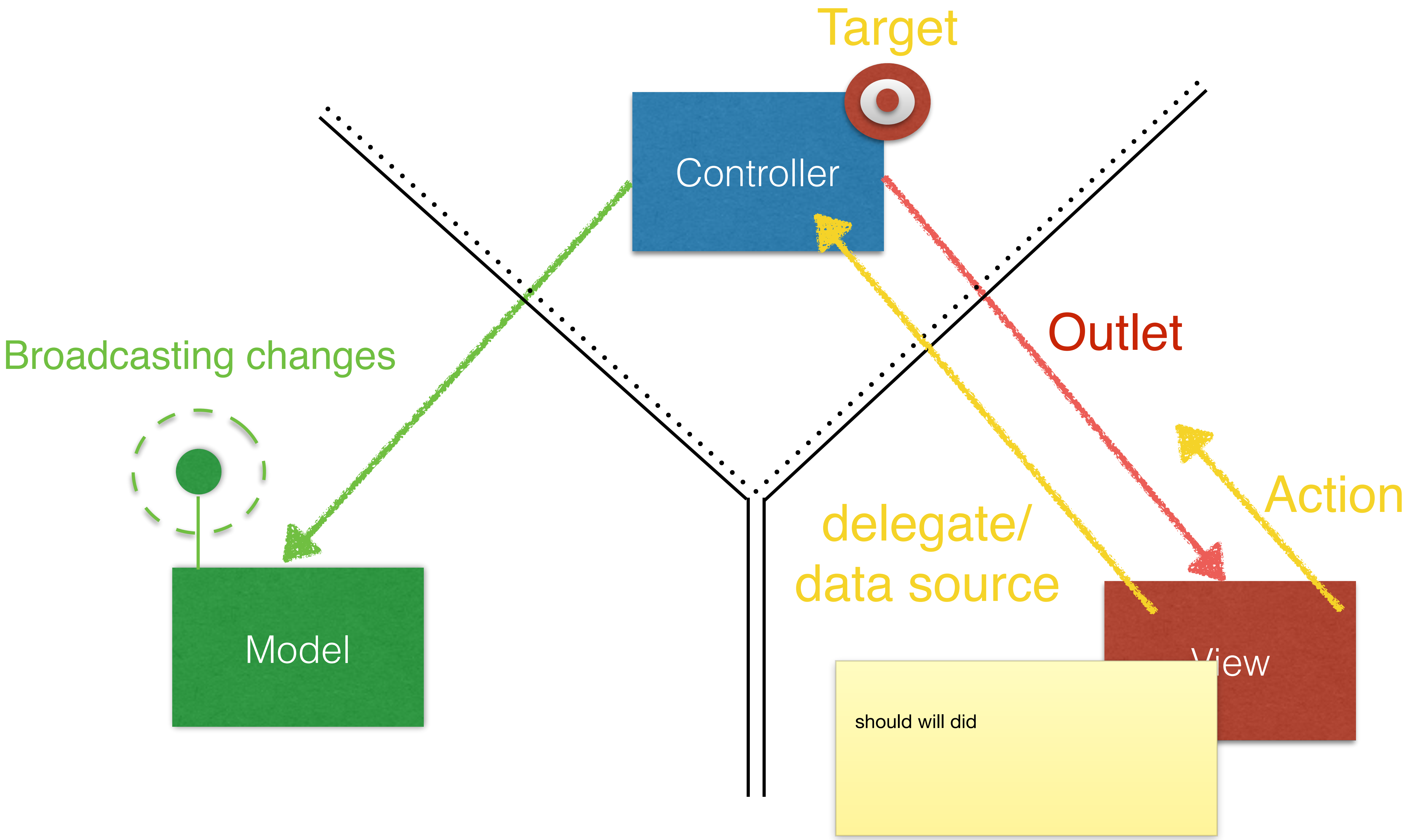
MVC

MVC

- **Model** is "what" is your app
- **Controller** is "how" data is displayed on screen
- **View** Building blocks of UI

MVC

It's all about data flow



should will did

Other patterns

- **MVP** (model, view, presenter)
- **MVVM** (model, view, view model)
- **Viper** (View, Interactor, Presenter, Entities, Router)

MWTF

Костыль Driven Development

Command line application “Hello, World!”

```objc
#import <Foundation/Foundation.h>

int main(int argc, const char * argv[]) {
    @autoreleasepool {
        // insert code here...
        NSLog(@"Hello, World!");
    }
    return 0;
}
```

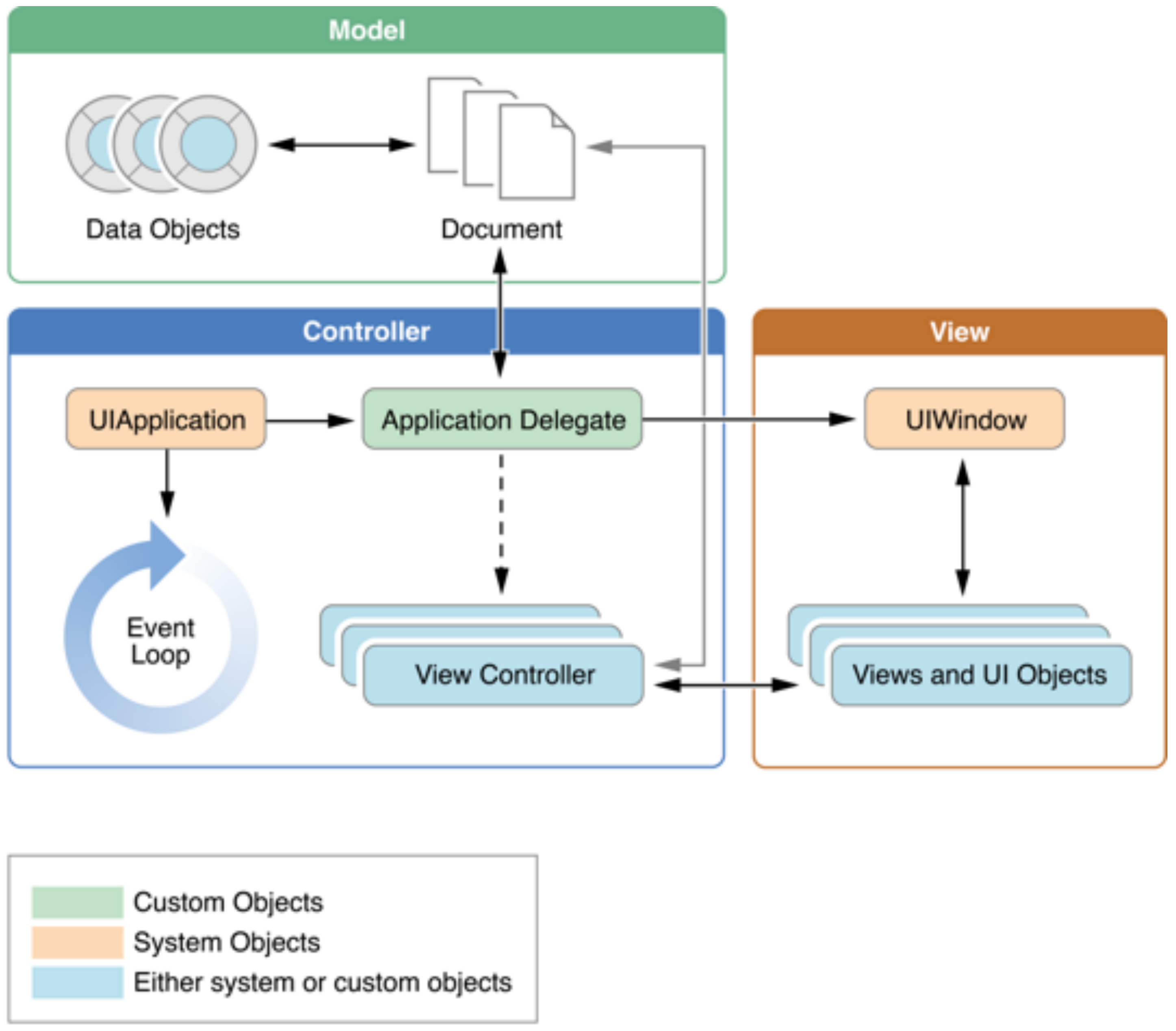
```
> Hello, World!
```

iOS application main() function

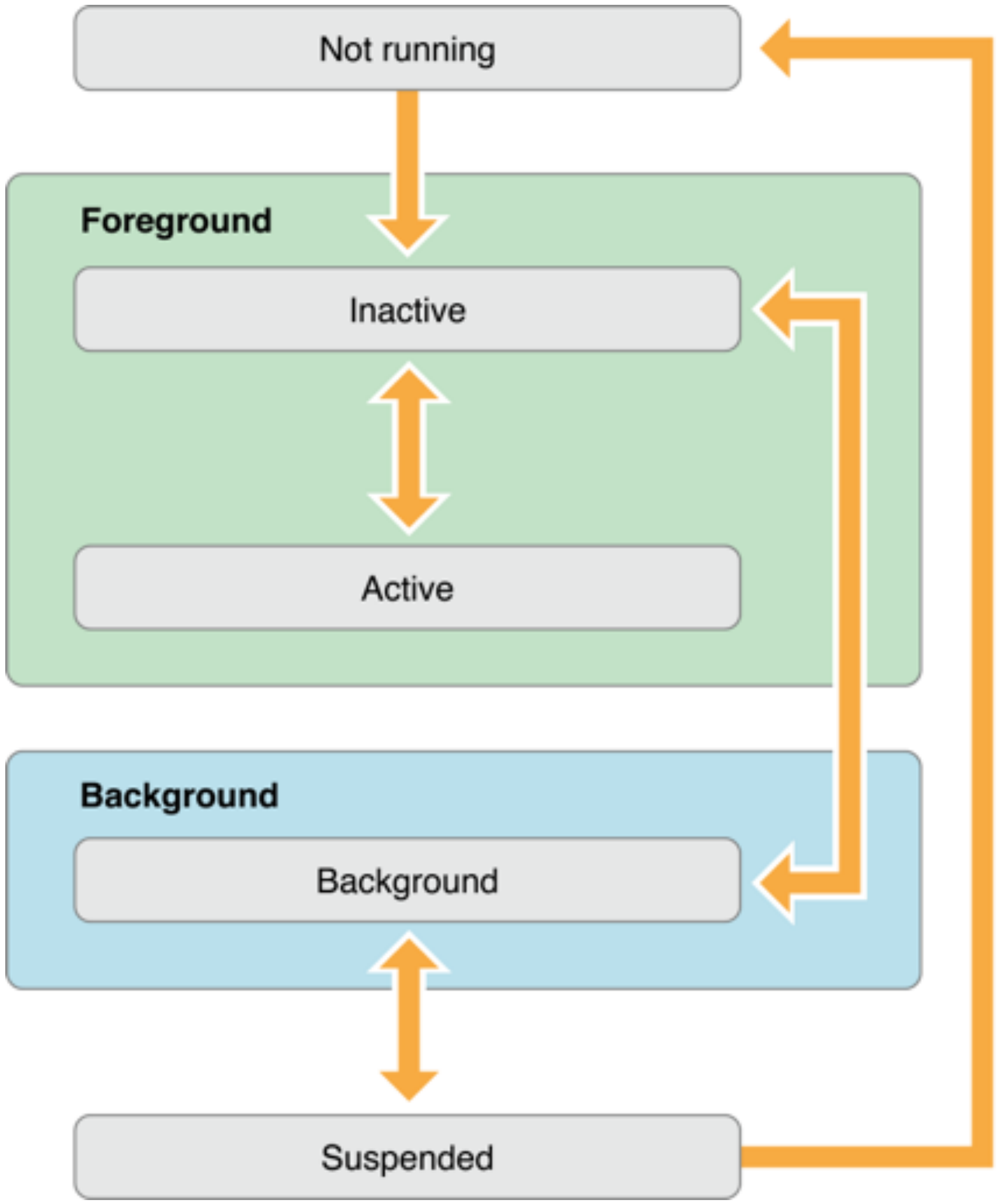
```
#import <UIKit/UIKit.h>
#import "AppDelegate.h"

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

- UIKit imported
- UIApplicationMain() function
- AppDelegate class name passed as string

Structure of an App



Application states

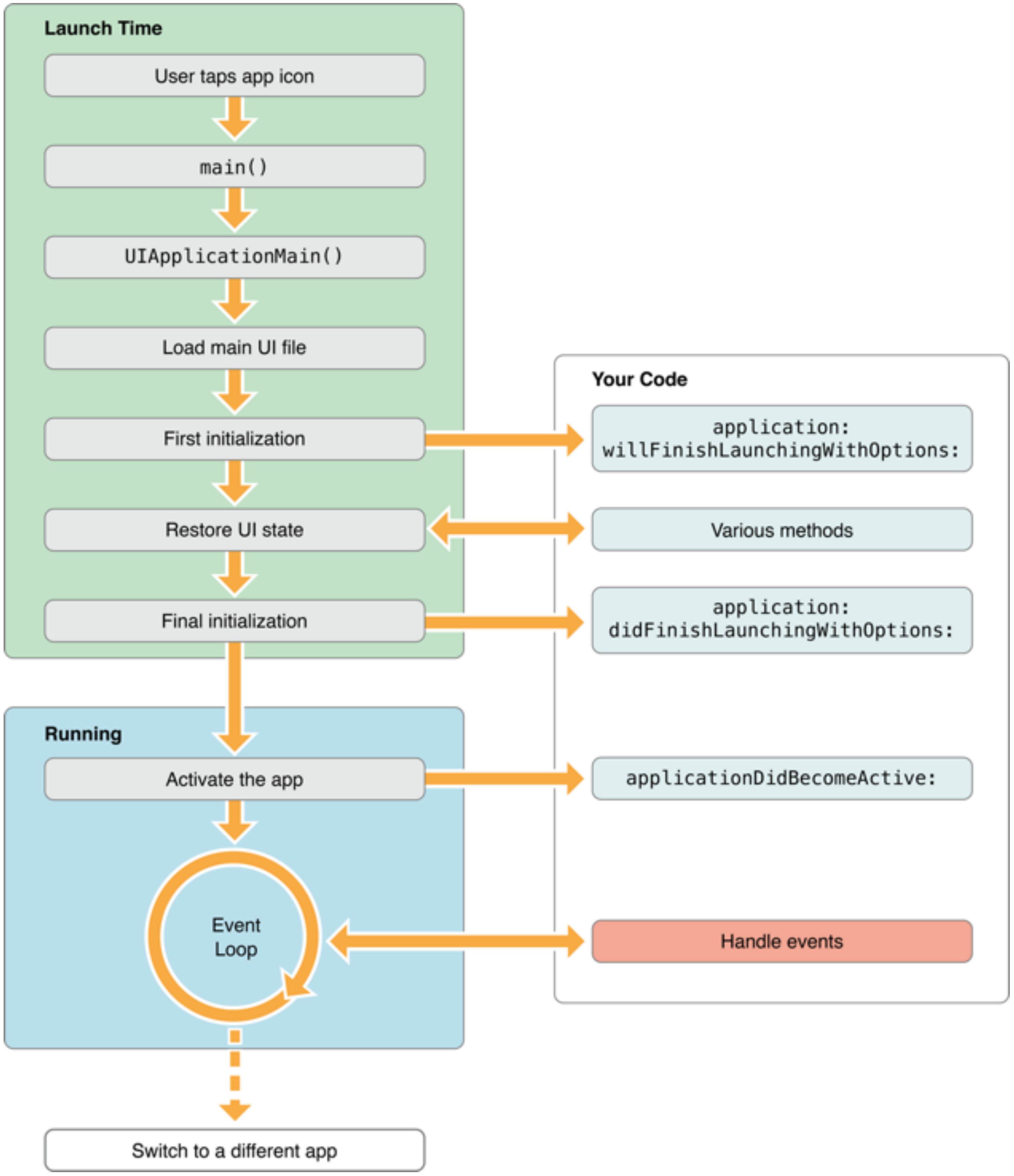


Managing State Transitions

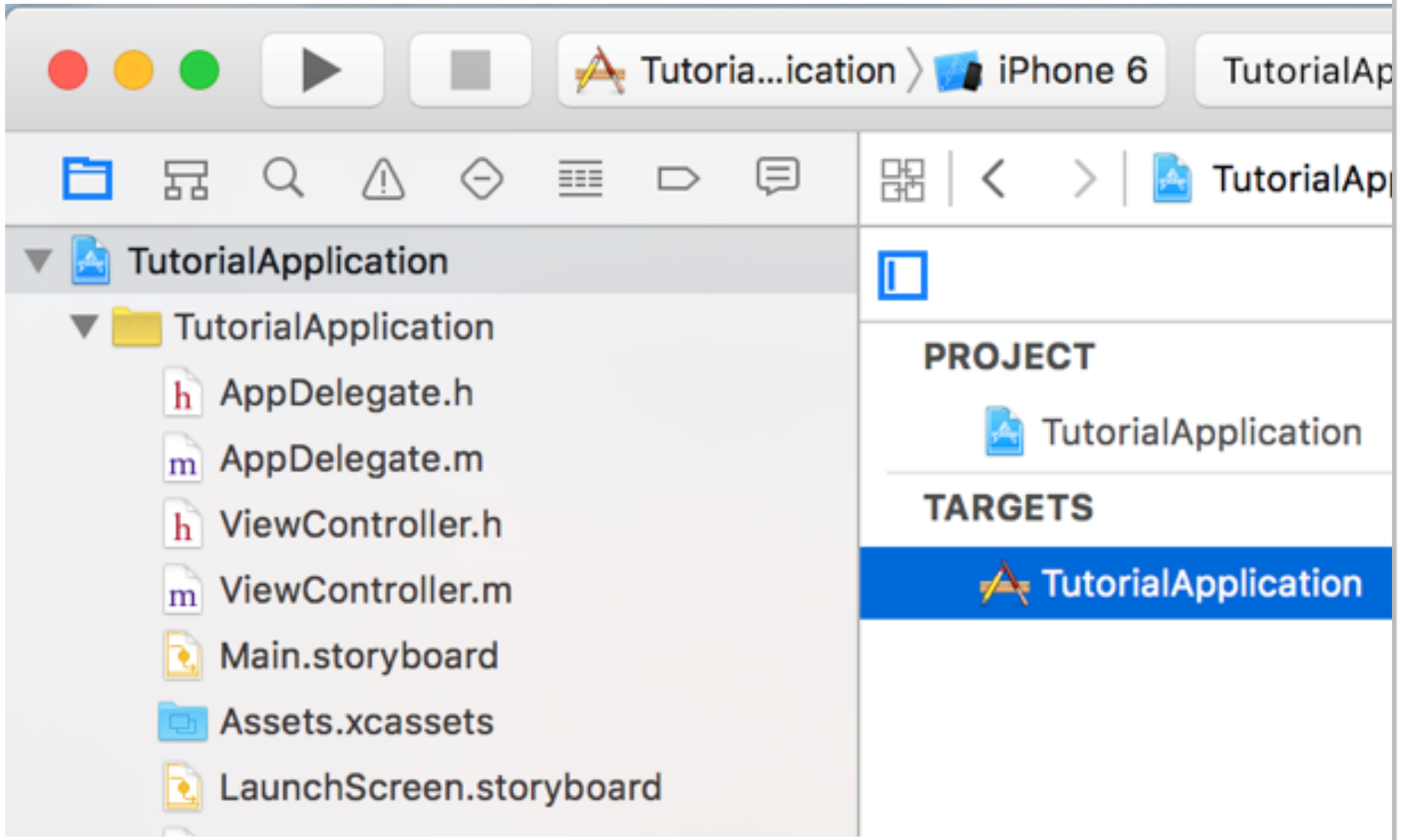
- Launch time
    - **application: willFinishLaunchingWithOptions:**
    - **application: didFinishLaunchingWithOptions:**
- Transitioning to and from active state
    - **applicationDidBecomeActive:**
    - **applicationWillResignActive:** (Called when leaving the foreground state.)
- Transitioning to foreground/background
    - **applicationDidEnterBackground:**
    - **applicationWillEnterForeground:** (Called when transitioning out of the background state.)
- Termination
    - **applicationWillTerminate:**

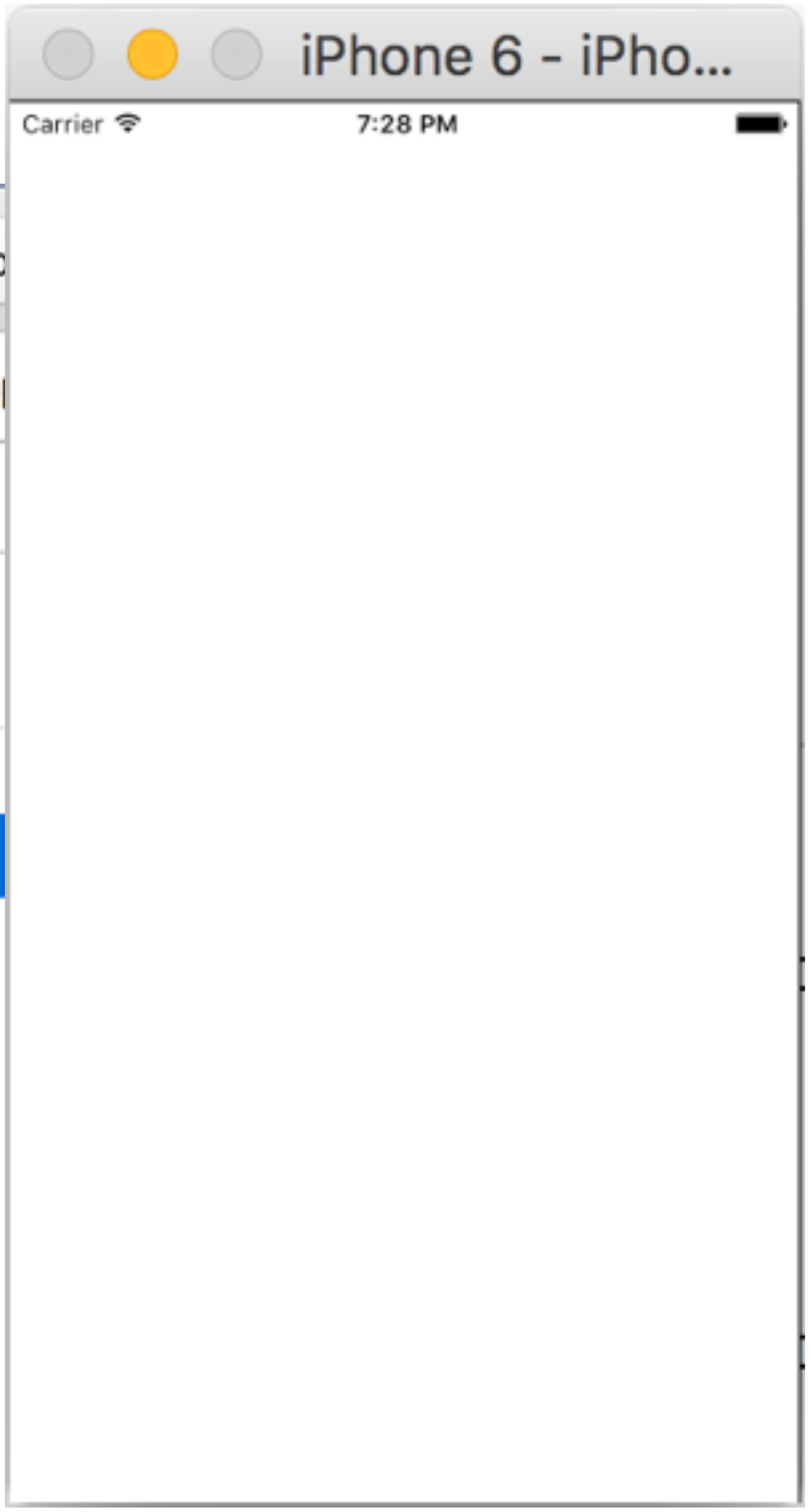
Other AppDelegate events

- **applicationDidReceiveMemoryWarning:**
- **application: openURL: options:**
- **application: didReceiveRemoteNotification: fetchCompletionHandler:**
- **applicationSignificantTimeChange:**

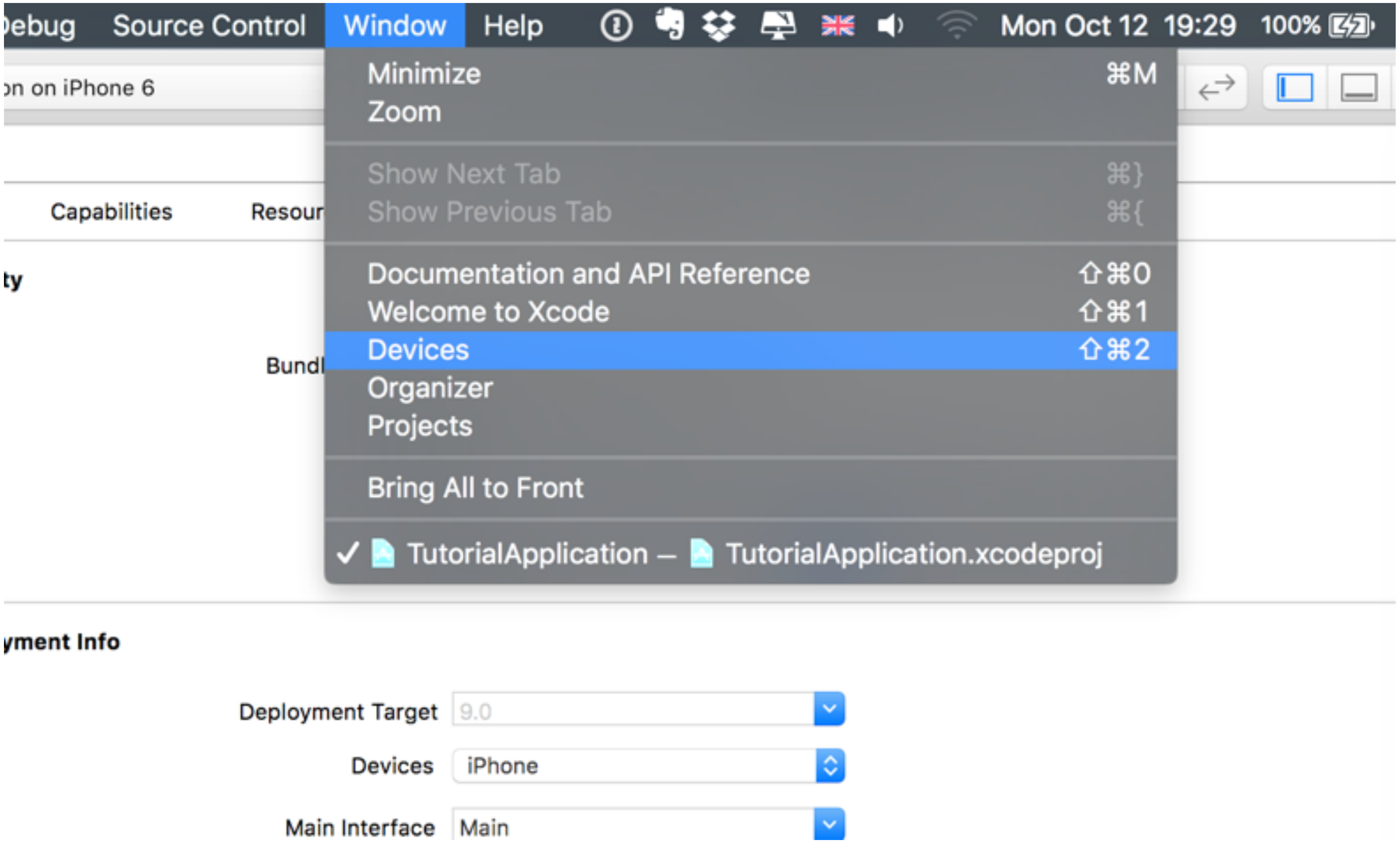

Launch Time
User taps app icon
main()
UIApplicationMain()
Load main UI file
First initialization
Restore UI state
Final initialization
Running
Activate the app
Event Loop
Switch to a different app
Your Code
application: willFinishLaunchingWithOptions:
Various methods
application: didFinishLaunchingWithOptions:
applicationDidBecomeActive:
Handle events

Running app on simulator





Managing simulators in Xcode 7

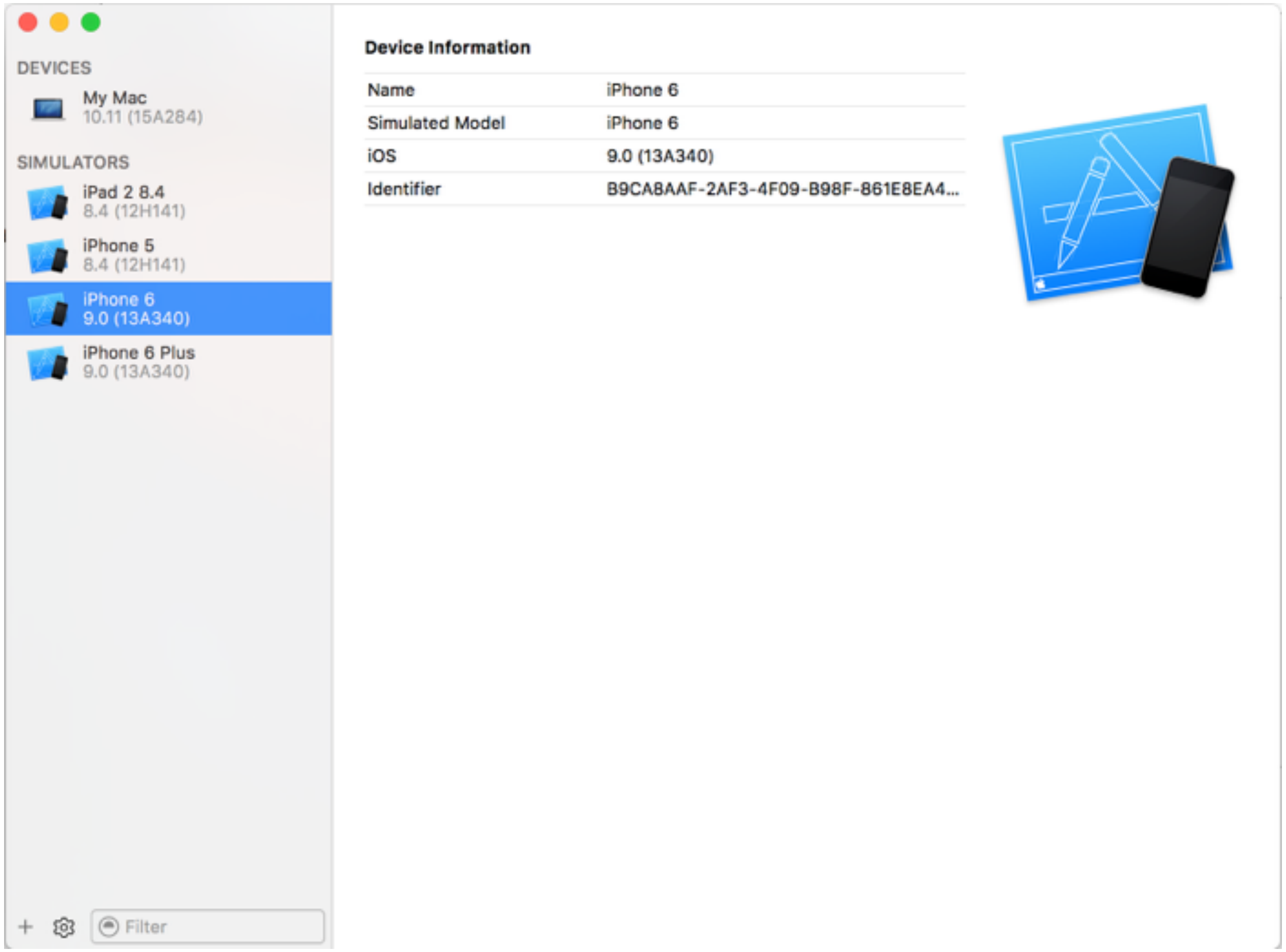


Deployment Target 9.0

Devices iPhone

Main Interface Main

Managing simulators in Xcode 7



DEVICES

My Mac
10.11 (15A284)

SIMULATORS

iPad 2 8.4
8.4 (12H141)

iPhone 5
8.4 (12H141)

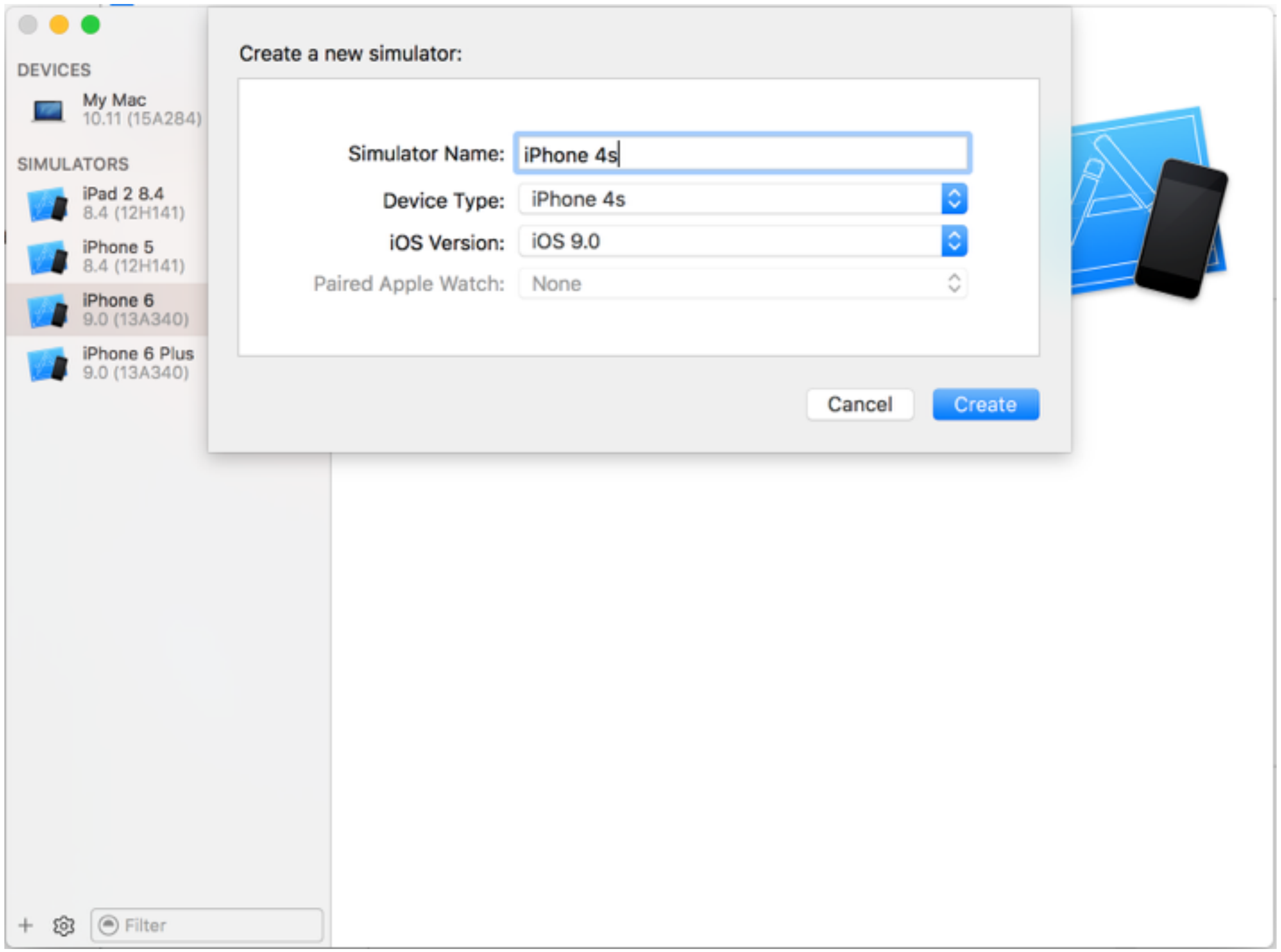
iPhone 6
9.0 (13A340)

iPhone 6 Plus
9.0 (13A340)

Device Information

Name	iPhone 6
Simulated Model	iPhone 6
iOS	9.0 (13A340)
Identifier	B9CA8AAF-2AF3-4F09-B98F-861E8EA4...

Managing simulators in Xcode 7



Create a new simulator:

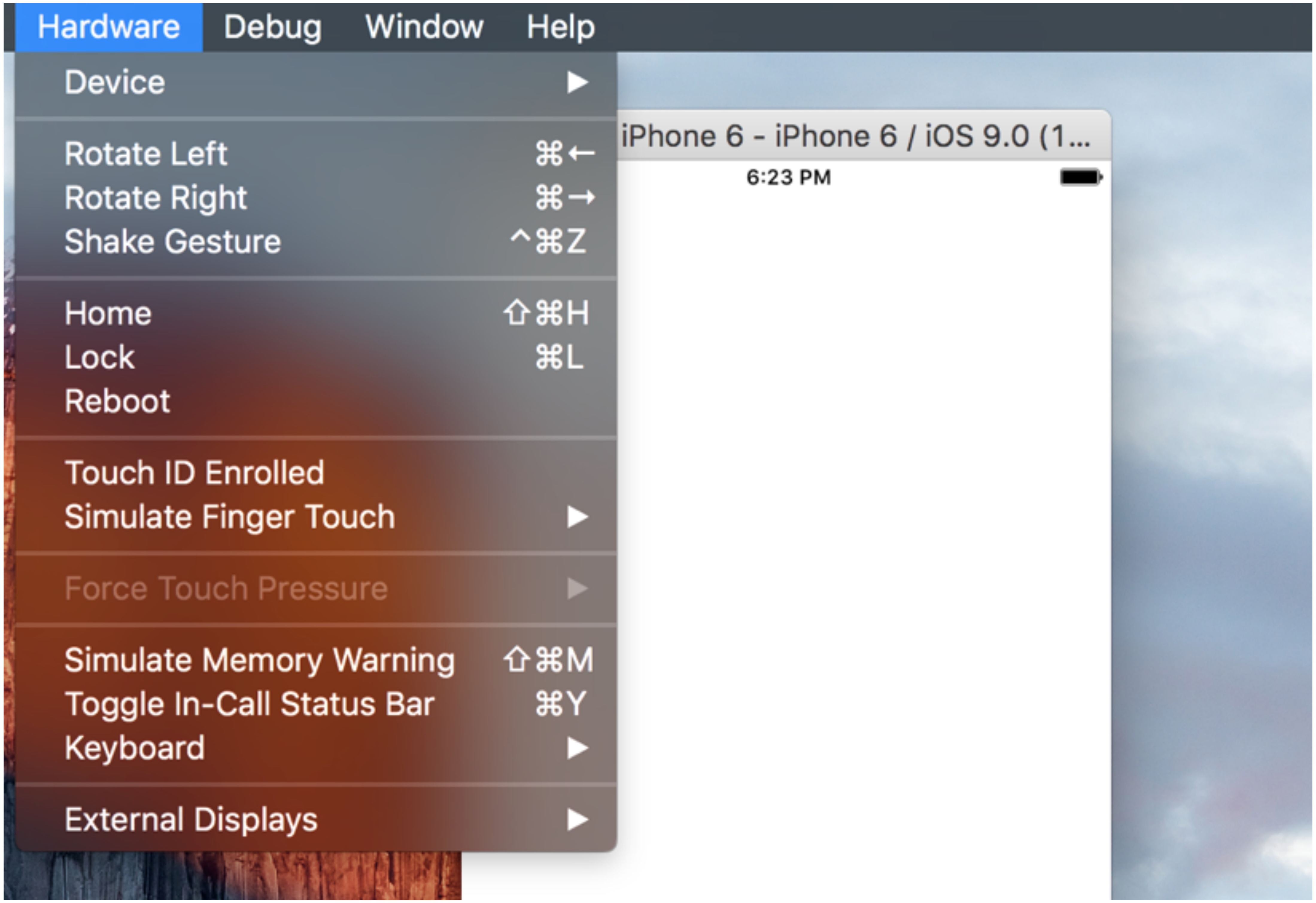
Simulator Name: iPhone 4s

Device Type: iPhone 4s

iOS Version: iOS 9.0

Paired Apple Watch: None

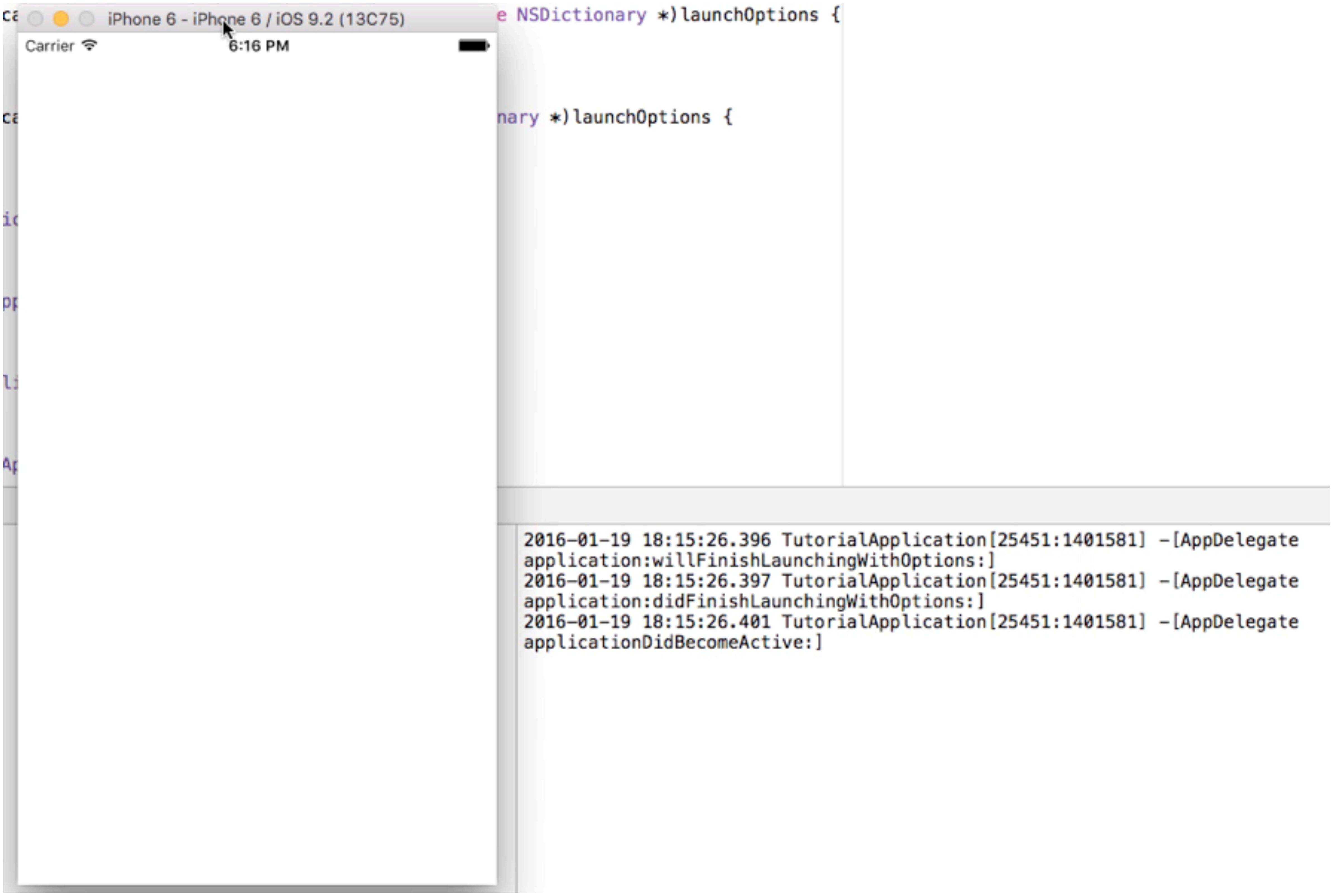
Simulating hardware events



Let’s execute some code!

```
@implementation AppDelegate
- (BOOL)application:(UIApplication *)application willFinishLaunchingWithOptions:(nullable NSDictionary
*)launchOptions {
    NSLog(@"%s", __PRETTY_FUNCTION__);
    return YES;
}

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions {
    NSLog(@"%s", __PRETTY_FUNCTION__);
    return YES;
}

- (void)applicationDidBecomeActive:(UIApplication *)application {
    NSLog(@"%s", __PRETTY_FUNCTION__);
}

- (void)applicationDidEnterBackground:(UIApplication *)application {
    NSLog(@"%s", __PRETTY_FUNCTION__);
}

- (void)applicationWillResignActive:(UIApplication *)application {
    NSLog(@"%s", __PRETTY_FUNCTION__);
}

- (void)applicationWillEnterForeground:(UIApplication *)application {
    NSLog(@"%s", __PRETTY_FUNCTION__);
}

- (void)applicationWillTerminate:(UIApplication *)application {
    NSLog(@"%s", __PRETTY_FUNCTION__);
}
@end
```

Let's execute some code!



```
e NSDictionary *)launchOptions {

nary *)launchOptions {
```

```
2016-01-19 18:15:26.396 TutorialApplication[25451:1401581] -[AppDelegate
application:willFinishLaunchingWithOptions:]
2016-01-19 18:15:26.397 TutorialApplication[25451:1401581] -[AppDelegate
application:didFinishLaunchingWithOptions:]
2016-01-19 18:15:26.401 TutorialApplication[25451:1401581] -[AppDelegate
applicationDidBecomeActive:]
```

NSUserDefaults

Used to store some preferences, flags, small amounts of data.

Don’t use it as a persistent store for large amounts of data on real projects!

Can store:

- NSData
- NSString
- NSNumber
- NSDate
- NSArray
- NSDictionary

NSUserDefaults

Storing data

```
NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];

 [defaults setInteger:1 forKey:@"age"];
 [defaults setFloat:5.5f forKey:@"float"];
 [defaults setBool:YES forKey:@"bool"];
 [defaults setObject:@YES forKey:@"boolWrappedInNumber"];
 [defaults setObject:@10 forKey:@"integerWrappedInNumber"];
 [defaults setObject:@"string" forKey:@"string"];

 [defaults synchronize];
```

Retrieving data

```
NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];
NSString *string = [defaults objectForKey:@“string"];

…
```

Note: calling synchronize after doing some changes is important!

NSCoding

NSObject <NSCoding> <-> NSData

- Use it when you need to store some object on the disk or in NSUserDefaults
- Only two methods required to serialise/deserialise object: **initWithCoder:** and **encodeWithCoder:**
- Don’t call them directly. Use **NSKeyedUnarchiver** and **NSKeyedArchiver** instead
- It isn’t used frequently, but sometimes it’s pretty useful

NSCoding example

```
@interface PMRParty : NSObject <NSCoding>

@property (nonatomic, readonly) NSString *partyID;
@property (nonatomic, readonly) NSString *name;
@property (nonatomic, readonly) NSDate *startDate;

@end

@implementation PMRParty

- (id)initWithCoder:(NSCoder *)aDecoder {
    self = [super init];
    if (self) {
        self.partyID = [aDecoder decodeObjectForKey:@"partyID"];
        self.name = [aDecoder decodeObjectForKey:@"name"];
        self.startDate = [aDecoder decodeObjectForKey:@"startDate"];
    }
    return self;
}

- (void)encodeWithCoder:(NSCoder *)aCoder {
    [aCoder encodeObject:self.partyID forKey:@"partyID"];
    [aCoder encodeObject:self.name forKey:@"name"];
    [aCoder encodeObject:self.startDate forKey:@"startDate"];
}

@end
```

NSCoding example

```
PMRParty *party = [[PMRParty alloc] init];
party.name = @"New Year Party 2016";
party.partyID = @"123";
party.startDate = [NSDate date];

NSArray *partiesArray = @[party];

NSData *data = [NSKeyedArchiver archivedDataWithRootObject:partiesArray];
[[NSUserDefaults standardUserDefaults] setObject:data forKey:@"parties"];

…

NSData *data = [[NSUserDefaults standardUserDefaults] objectForKey:@"parties"];
NSArray *partiesArray = [NSKeyedUnarchiver unarchiveObjectWithData:data];
```

Homework

- Create an iOS application and print into console all state transition events mentioned in this lecture

- Create an object upon each event with properties “eventName”, “eventDate” and also an unique “eventID” will be great, google how to generate it ;)

- Store these objects in array, serialise it to NSData and save it in NSUserDefaults when home button is pressed

- Restore array when app will start next time

Useful links

- https://developer.apple.com/library/ios/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/TheAppLifeCycle/TheAppLifeCycle.html
- https://developer.apple.com/library/prerelease/ios/documentation/UIKit/Reference/UIApplicationDelegate_Protocol/index.html
- http://nshipster.com/nscoding/
- http://nshipster.com/nsnotification-and-nsnotificationcenter/

Thank you for attention!

mail: mikhail.kramskoy@softheme.com
skype: wirrwarr74