7CC003 Distributed And Mobile Computing

IOS

Apple iPhone

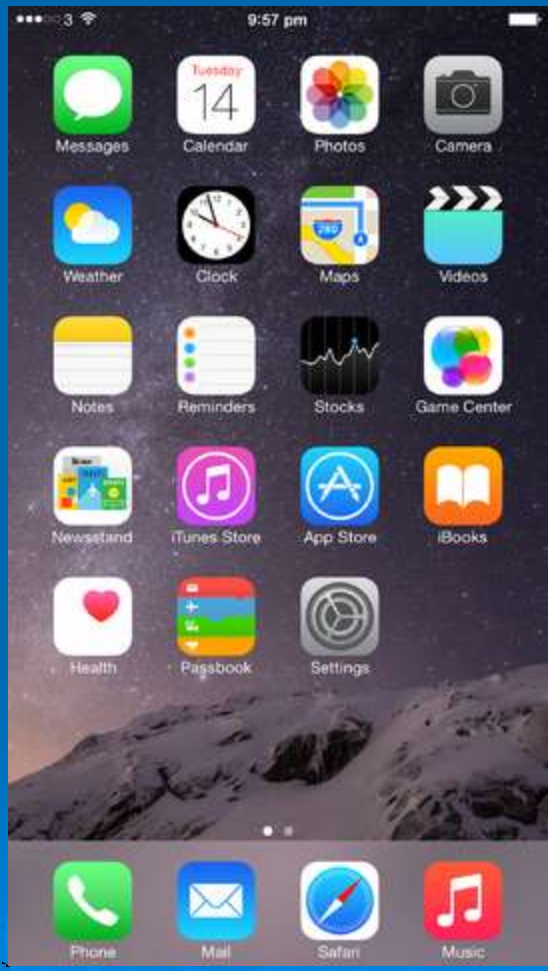
- Smartphone designed and marketed by Apple Inc.
- First multi-touch smartphone to be designed for finger operation rather than stylus.
- First released on June 29, 2007.
- Current model is iPhone 6

History

- 2007 - iPhone 1st gen, iPod Touch 1st gen
- 2008 - iPhone 3G, iPod Touch 2nd gen
- 2009 - iPhone 3GS, iPod Touch 3rd gen
- 2010 - iPhone 4, iPod Touch 4th gen, iPad
- 2011 - iPhone 4S, iPod Touch 4th gen, iPad 2
- 2012 - iPhone 5, iPod Touch 5th gen, New iPad
- 2013 - iPhone 5C, 5S, iPad Air
- 2014 - iPhone 6/6 Plus , iPad Air 2

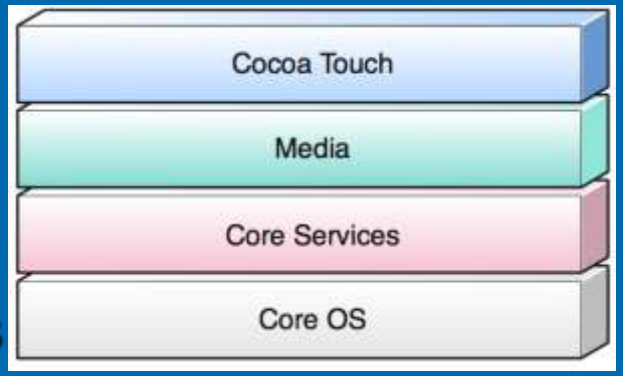
What is IOS?

- iOS is the operating system that runs on iPad, iPhone, and iPod touch devices.
- XNU-based – derived from NeXTSTEP, with elements from BSD and Mach
- The operating system manages the device hardware and provides the technologies required to implement native apps.



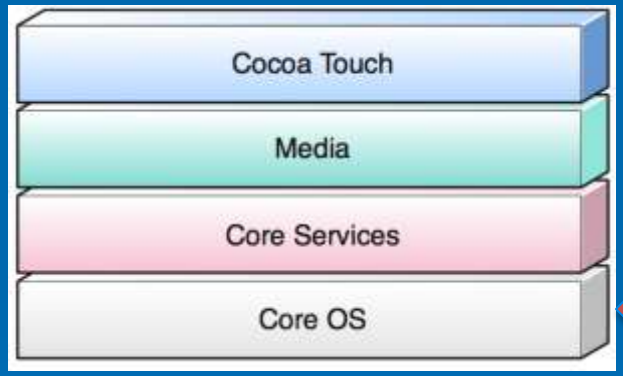
iOS Architecture

- Implemented as a number of layers
- Lower layers provide fundamental services and technologies
- Higher layers provide more sophisticated services
  - Builds upon the functionality provided by the lower layers
  - Provides object-oriented abstractions for lower layer constructs
- Each layer has a number of frameworks (packages of system interfaces)
  - Each framework contains dynamically shared libraries and associated resources (header files, images, etc)
  - When a framework is used, they need to be linked into the project



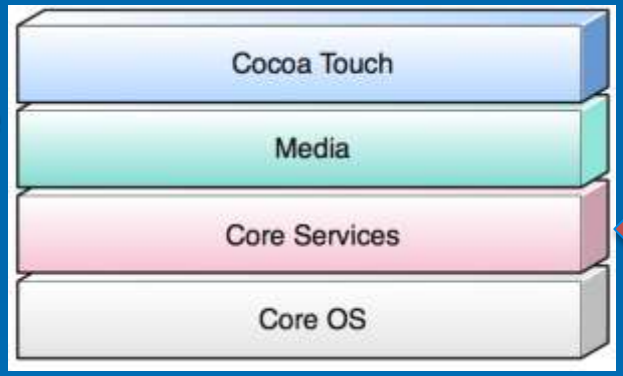
iOS Overview: CoreOS

- System Framework (based on Mach)
  - Threading (POSIX)
  - Networking (BSD sockets)
  - File system
  - Service discovery (Bonjour & DNS)
  - Memory management
  - Math computations
- Core Bluetooth Framework and External Accessory Framework
  - Support for communicating with hardware accessories
- Accelerate Framework
  - DSP, linear algebra and image processing optimized for hardware
- Security Framework
  - Crypto library and keychain Services (secure storage of passwords, keys



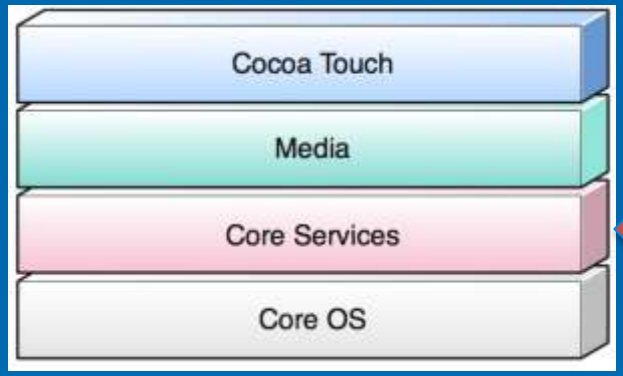
iOS Overview: Core Services

- High level features
  - iCloud storage
  - Automatic reference counting
  - SQLite: lightweight SQL database
  - Grand Central Dispatch (GCD): manage concurrent execution of tasks
    - Thread management code moved to the system level
    - Tasks specified are added to an appropriate dispatch queue
  - Block objects: a C-level language construct; an anonymous function and the data (a closure or lambda)
  - In-App purchase: process financial transactions from iTune account
  - XML support



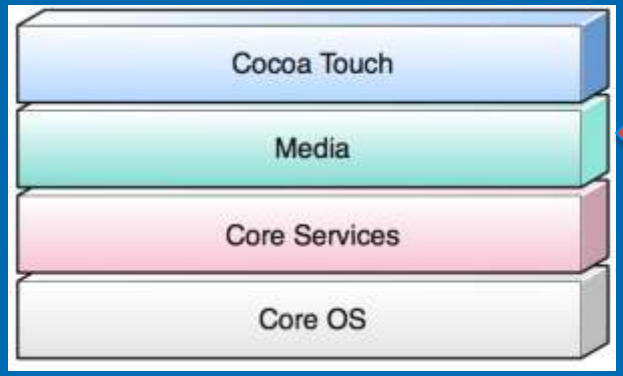
iOS Overview: Core Services

- CFNetwork Framework
  - Object-oriented abstractions for working with network protocols (DNS, http, Bonjour services)
- Core Telephony Framework
- System Configuration Framework
  - Determine network configuration
- Social Framework
  - Post status updates and images to social networks
- Foundation Framework: objective-C wrapper
- Address Book Framework
- Core Data Framework
- Core Foundation Framework
- Core Media Framework: C interface for media
- Core Location Framework
- Newsstand Kit Framework
- Store Kit Framework: in app purchase



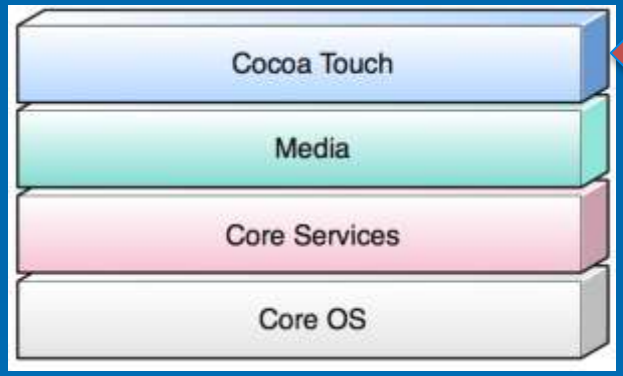
iOS Overview: Media

- Graphics
  - Core graphics framework
  - Core animation framework
  - Core image framework
  - OpenGL ES and GLKit framework
  - Core text framework
- Audio/video
  - Meida player framework: access to iTunes
  - OpenAL framework: positional audio playback
  - Core audio framework: Airplay, recording audio
  - Core video framework: buffer support for core media framework
  - AV Foundation framework (Objective-C interface): playback, recording, Airplay
  - Asset Library Framework: retrieving photos and videos from user's device



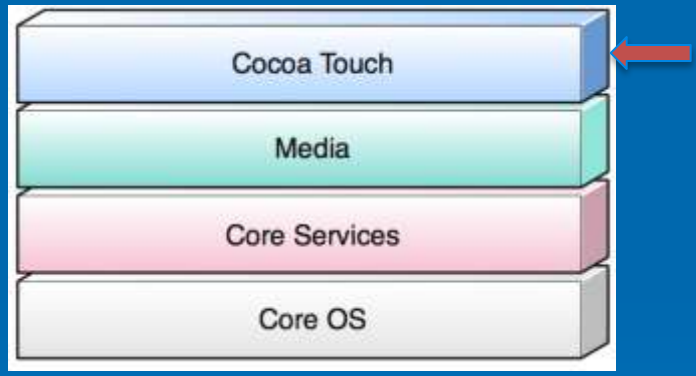
iOS Overview: Cocoa Touch

- UI Kit Framework
  - Apple push notification service
  - Storyboards: supplant nib files as the recommended way to design your application's user interface
  - Document Support: UIDocument class for managing the data associated with user documents
  - Multitasking
  - Printing: support allows applications to send content wirelessly to nearby printers
  - Local push notification
  - Gesture recognizers
  - Accelerometer data, built-in camera, battery state information, proximity sensor information



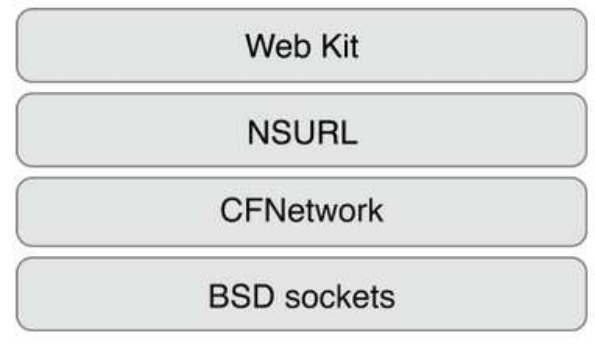
iOS Overview: Cocoa Touch (Cont'd)

- Game Kit Framework
  - Peer-to-peer services: over Bluetooth, e.g. multi-player games
- Address Book UI Framework: contact management
- iAd Framework: deliver banner-based advertisements from your application
- Map Kit Framework: a scrollable map interface
- Message UI Framework: support for composing and queuing email messages in the user's outbox



Networking

- Core Services framework provides a library of abstractions for network protocols.
  - Working with BSD sockets
  - Creating encrypted connections using SSL or TLS
  - Resolving DNS hosts
  - Working with HTTP, authenticating HTTP and HTTPS servers
  - Working with FTP servers
  - Publishing, resolving and browsing Bonjour services: CFNetServices API provides access to Bonjour



iCloud

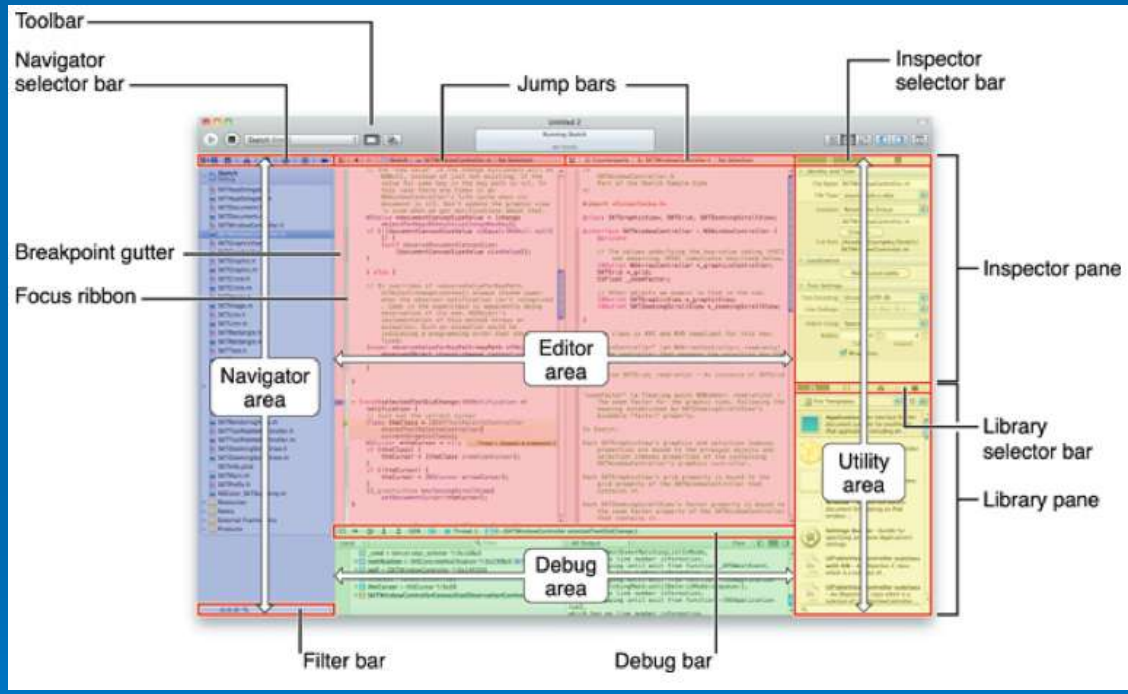
Fundamentally: nothing more than a URL of a shared directory

- Two storage models
  - iCloud document storage: store user documents and app data in the user's iCloud account
  - iCloud key-value data storage: share small amounts of noncritical configuration data among instances of your app

- iCloud-specific entitlements required
  - Select app target in Xcode
  - Select the Summary tab
  - In the Entitlements section, enable the Enable Entitlements checkbox

Xcode

- The IDE for developing MacOSX and iOS applications
  - Single window, supporting multiple workspace
  - Integrated Interface Builder
  - Assistant Editor (split pane that loads related files, such as header files etc)
  - Dynamic syntax checking and alert
  - Version editor with Git or Subversion integration
  - LLVM 2.0 editor with support for C, C++ and Objective-C
  - LLDB debugger



iOS Software Development Kit

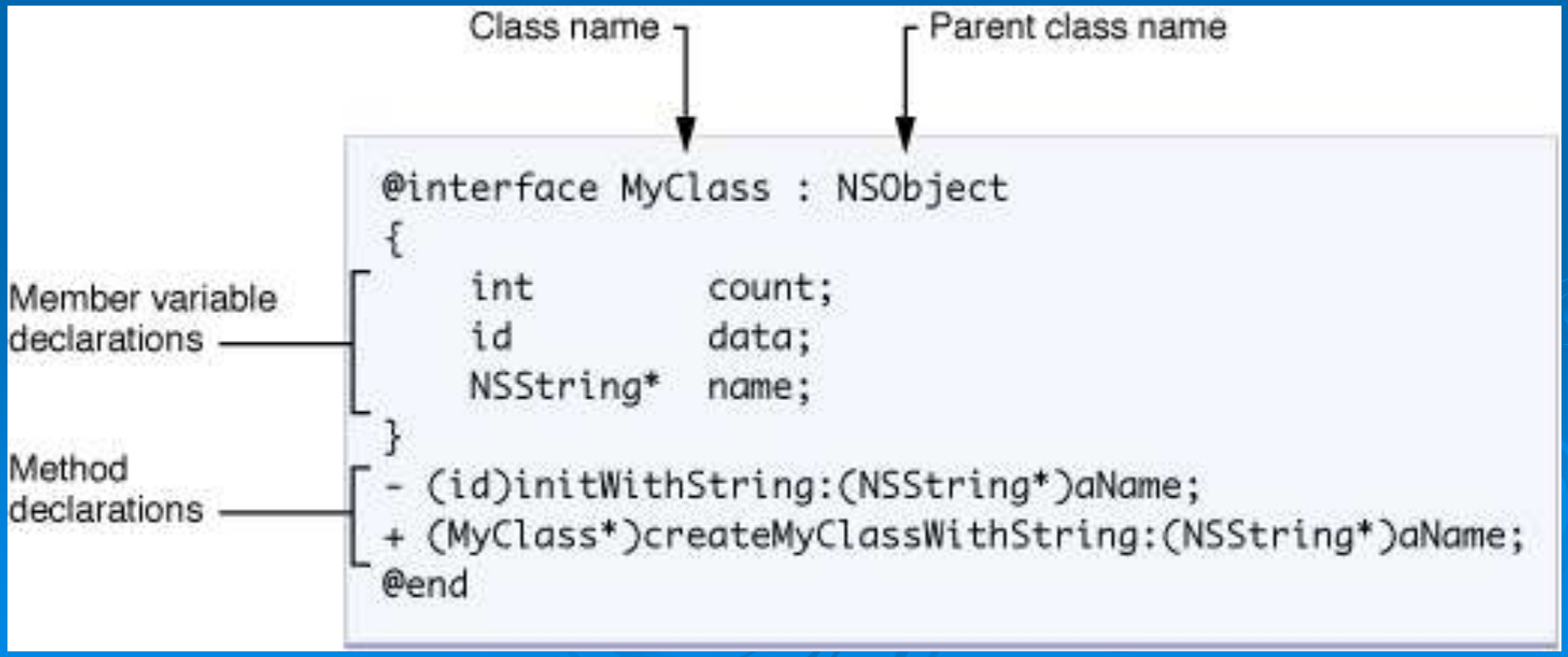
- First released on March 6, 2008
- New SDK released with each new iOS
- Requires an iOS Developer Program fee at US$99.00 per year to deploy to user devices
- Apps can only be published via Apple's App Store

Objective-C

- A strict superset of ANSI C
- Originally used within NeXT's NeXTSTEP OS
- Single inheritance
- Dynamic runtime: everything is looked up and dispatched at run time
- No garbage collection on iPhone, iTouch and iPad
- New types
  - id type: dynamic type to refer to any object
  - Selectors: refers to the identifier for a method after compilation

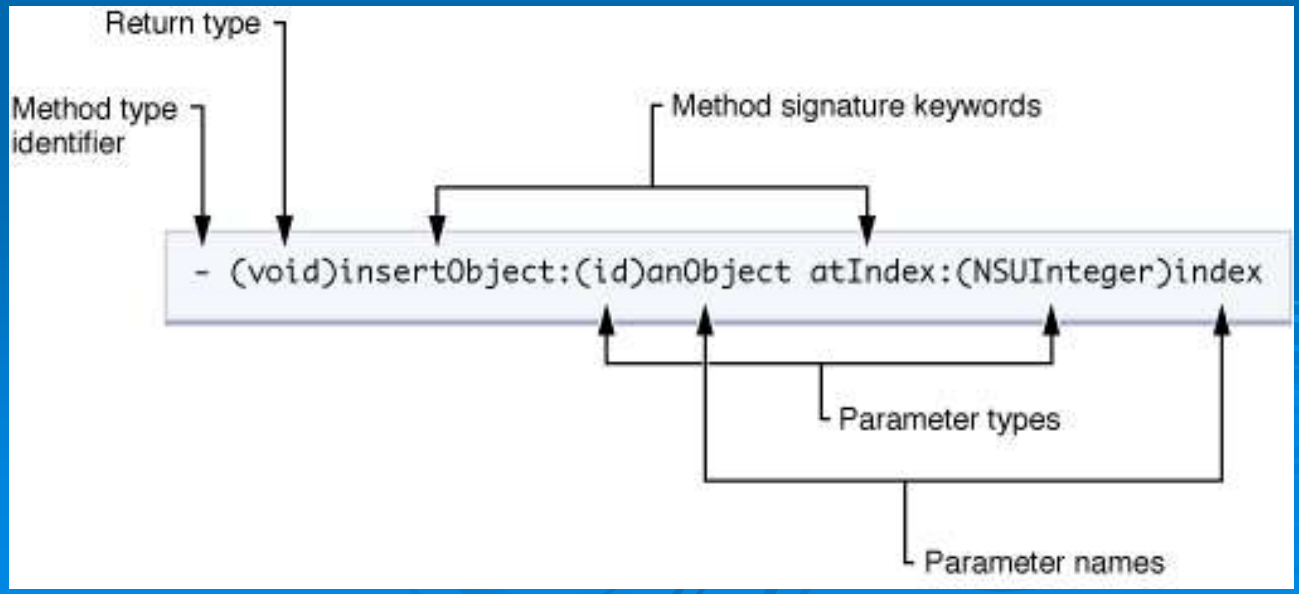
Classes

- Objective C is object-oriented
- Classes specified using an interface and an implementation



Methods

- A class in Objective-C can declare two types of methods: instance methods and class methods.
  - An instance method is a method whose execution is scoped to a particular instance of the class.
  - Class methods, by comparison, do not require you to create an instance (i.e. static methods)



Messaging

- When you want to call a method, you do so by **messaging** an object.
  - A message is the method signature, along with the parameter information the method needs.
  - All messages you send to an object are dispatched dynamically.
- Messages are enclosed by brackets ([and]).
  - Inside the brackets, the object receiving the message is on the left side and the message (along with any parameters required by the message) is on the right.

Messaging Example

- For example, to send the insertObject:atIndex: message to an object in the myArray variable, you would use the following syntax:

```
[myArray insertObject:anObject atIndex:0];
```

- Objects can be nested:

```
[[myAppObject theArray] insertObject:[myAppObject objectToInsert] atIndex:0];
```

- Obj-C also supports dot syntax:

```
myAppObject.theArray = aNewArray;
```

Objective-C Example

```
#import "Stack.h"

@implementation Stack
@synthesize numStack = _numStack;

- (NSMutableArray *) numStack {
    if (_numStack==nil)
        _numStack = [[NSMutableArray alloc] init];
    return _numStack;
}

- (void) push:(double)num {
    [self.numStack addObject:[NSNumber numberWithDouble:num]];
}

- (double) pop {
    NSNumber *numObject = [self.numStack lastObject];
    if(numObject) [self.numStack removeLastObject];
    NSLog(@"poped %@",numObject);
    return [numObject doubleValue];

}
@end
```

Objective-C stack.m file

@synthesize creates getter and setter methods
alloc: a class method

Method syntax
self: the instance itself
dot notation to access setter and getter method

C++ Implementation

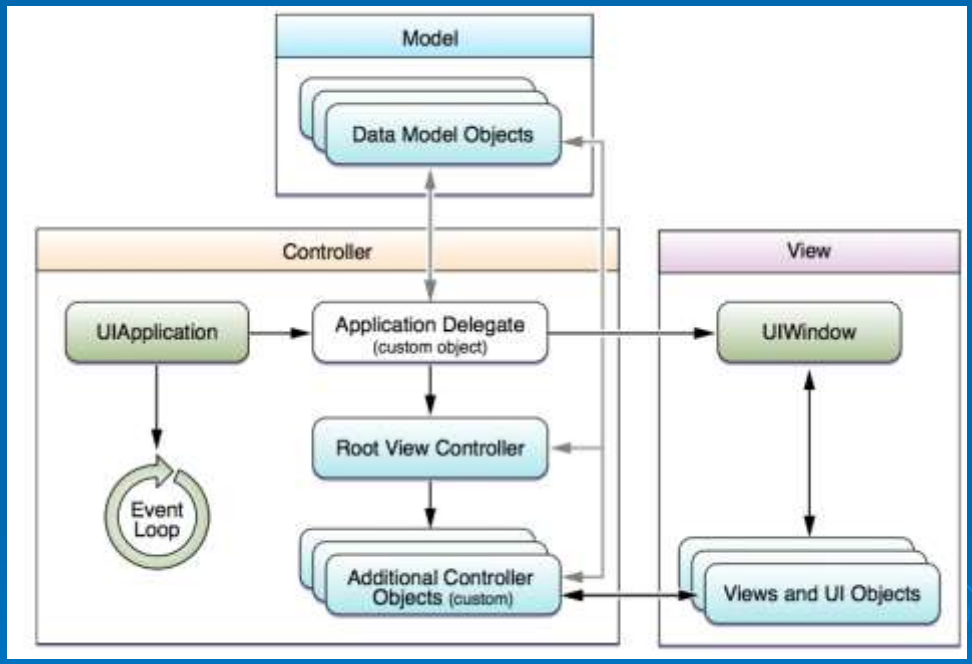
```
#include "stack.h"
#define stackSize 100

Stack::Stack()
{
    numStack = new double[stackSize];
    top = 0;
}

void Stack::push(double x)
{
    if(!is_full())
        num[top++] = x;
}

double Stack::pop()
{
    if(!is_empty())
        return num[--top];
    else
        return -1;
}
```

Method syntax

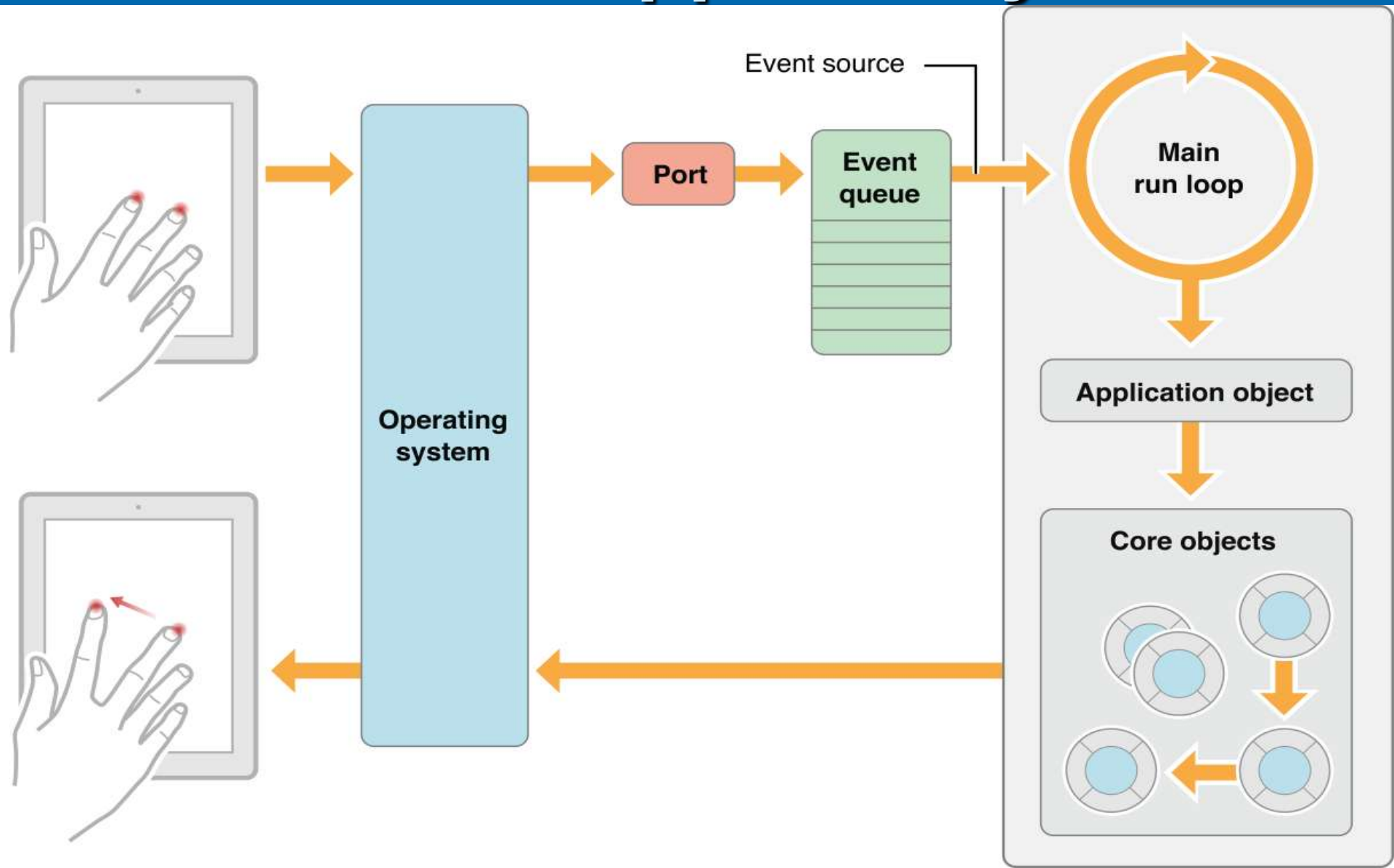
Model View Controller (MVC)

Key objects in iOS apps

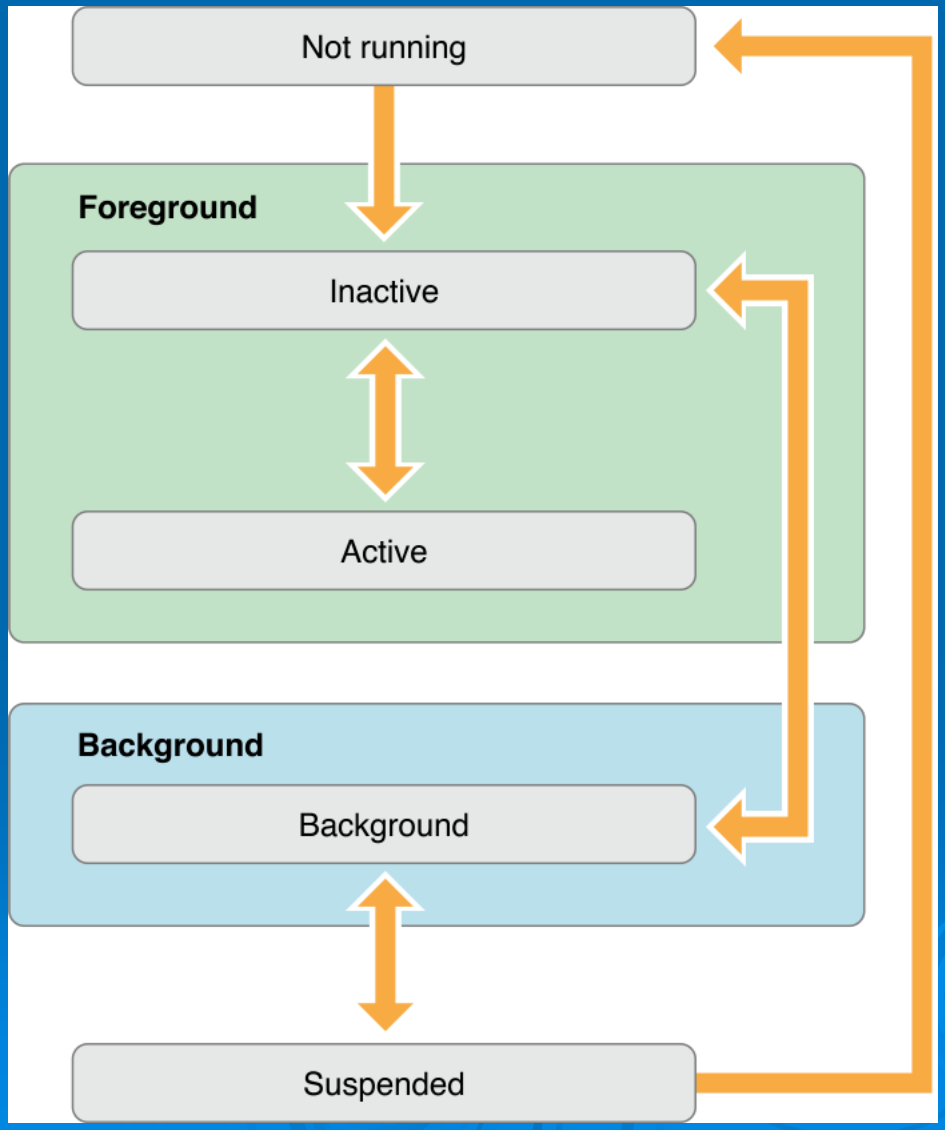
- UIApplication controller object:
  - manages the app event loop
  - coordinates other high-level app behaviors
  - custom app-level logic resides in your app delegate object
- AppDelegate custom object: created at app launch time, usually by the UIApplicationMain function
  - handle state transitions within the app



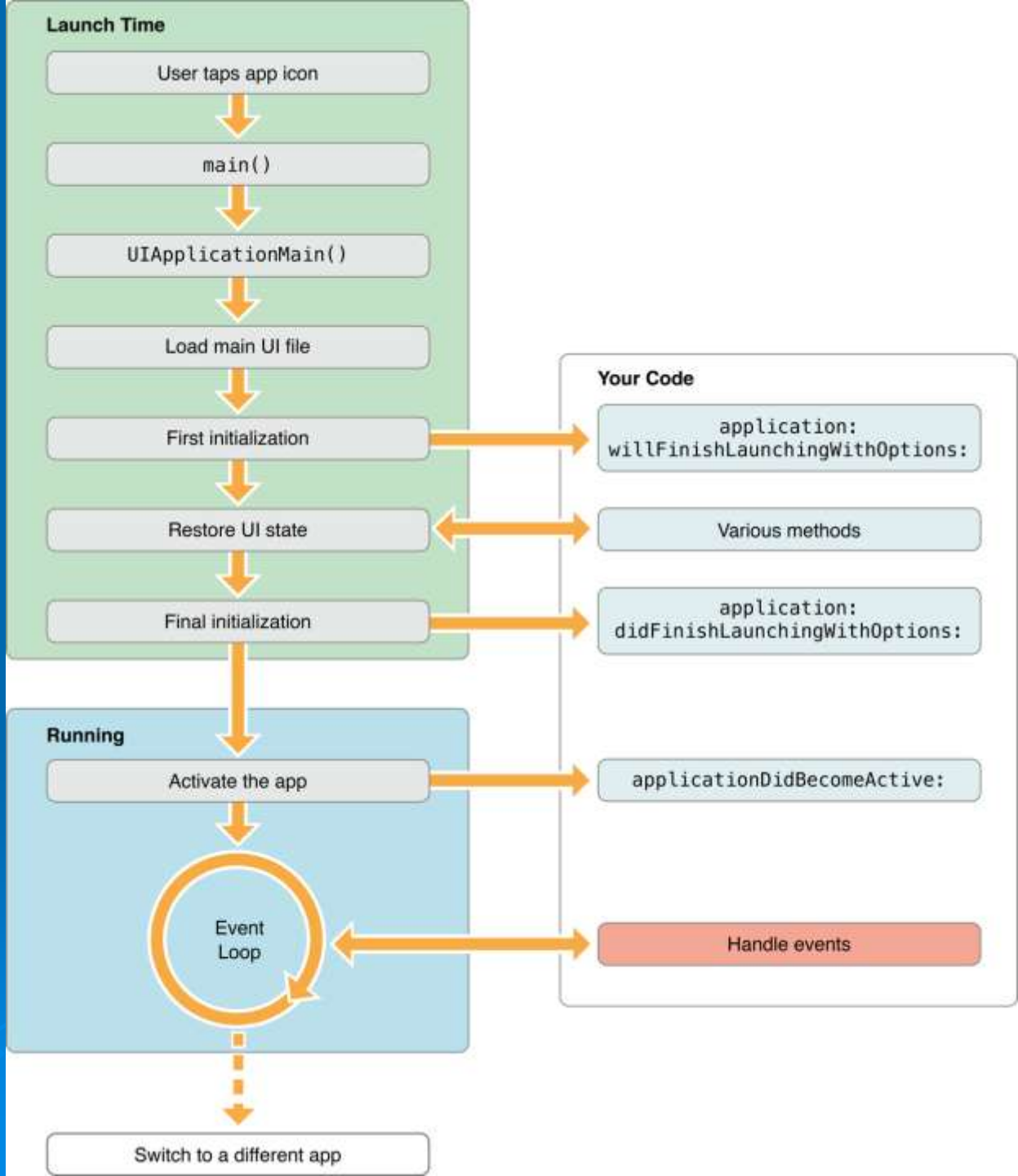
The iOS App Life Cycle



Execution States for Apps



Launching an App



App Termination

- Apps must be prepared for termination to happen at any time
- System-initiated termination is a normal part of an app's life cycle, usually to reclaim memory, may also terminate apps that are misbehaving.
- App will get a notification that it is about to be terminated.
- Suspended apps receive no notification when they are terminated.

- Users can terminate the app explicitly using the multitasking UI.
- User-initiated termination has the same effect as terminating a suspended app. The app's process is killed and no notification is sent to the app.