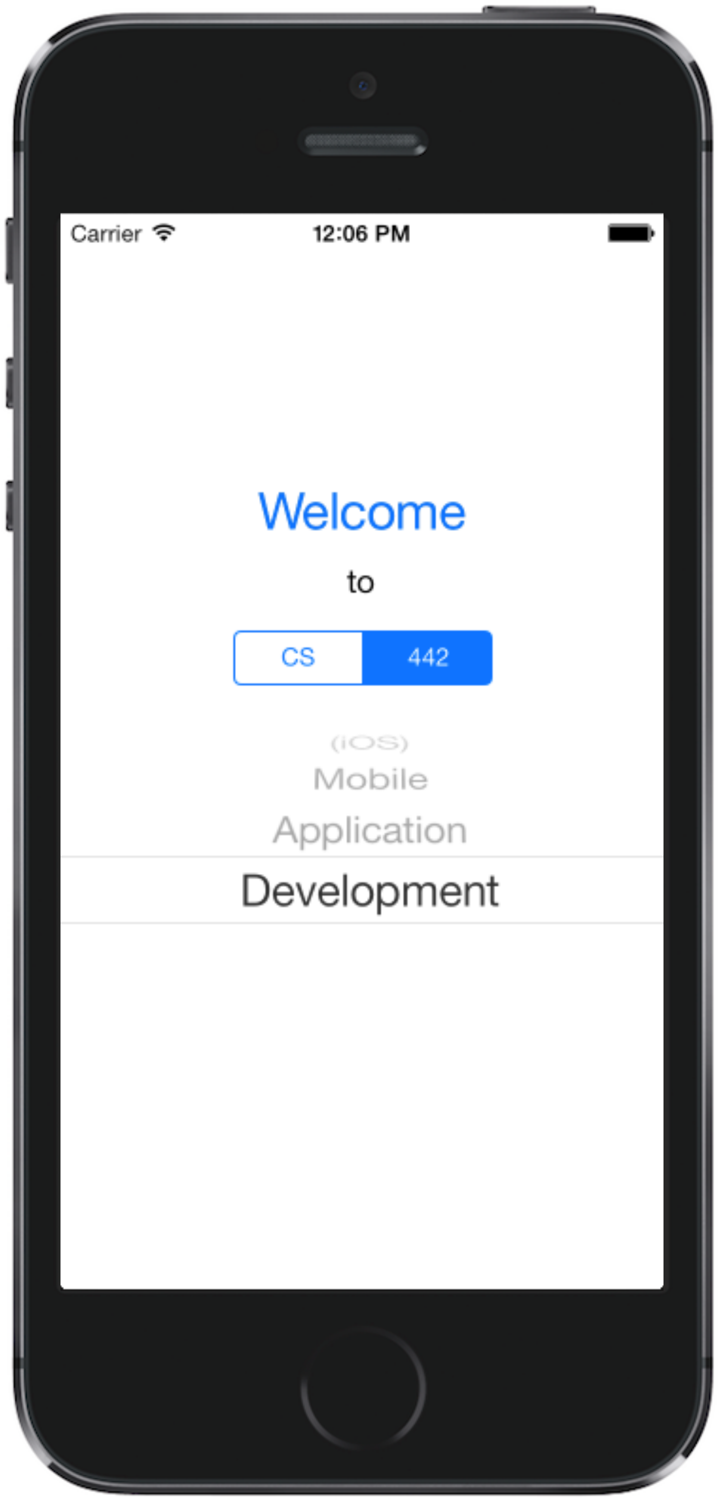

Carrier 12:06 PM

Welcome

to

CS 442

(iOS)
Mobile
Application
Development

Introductions

Science

Computer Science

CS 442: Mobile App Development

Michael Saelee <lee@iit.edu>

IIT College of Science
ILLINOIS INSTITUTE OF TECHNOLOGY

Michael (Sae) Lee

- lee@iit.edu
- moss.cs.iit.edu
- Office: SB 226A
    - Hours: MW, 11:30AM-1:30PM

Agenda

- Syllabus & Administrivia
- Course overview

Android section!

- Second CS 442 section
- This section = iOS; Section 2 = Android
  - Taught by industry instructor
  - No cross-attendance or assignment submission!

§Syllabus

Prerequisites

- “substantial” programming experience
  - previously, C was advantageous; no more!
- data structures (CS 331)
- systems programming (CS 351)
- databases (CS 425)

Prerequisites

- familiarity with Macs not needed (but handy)
- essentials:
  - yes, there is a right click
  - command (⌘) for control
  - single menu bar

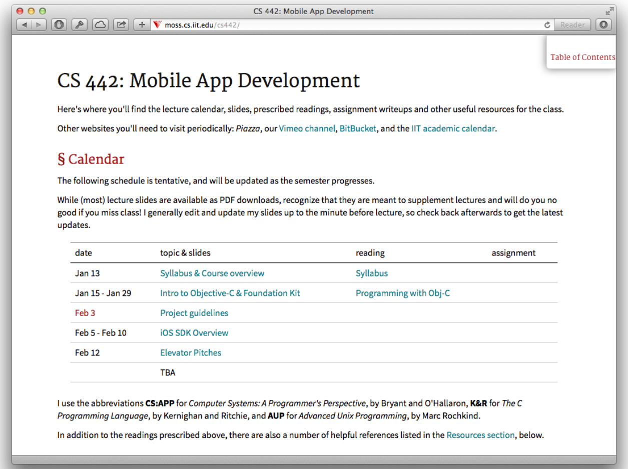
Online resources

1. Course website
   moss.cs.iit.edu/cs442

   - static information

     - syllabus, lecture calendar, assignments, slides, links to reading material

     - *not yet updated for Spring 2015!*



CS 442: Mobile App Development

moss.cs.iit.edu/cs442/

Table of Contents

CS 442: Mobile App Development

Here's where you'll find the lecture calendar, slides, prescribed readings, assignment writeups and other useful resources for the class.

Other websites you'll need to visit periodically: *Piazza*, our Vimeo channel, BitBucket, and the IIT academic calendar.

§ Calendar

The following schedule is tentative, and will be updated as the semester progresses.

While (most) lecture slides are available as PDF downloads, recognize that they are meant to supplement lectures and will do you no good if you miss class! I generally edit and update my slides up to the minute before lecture, so check back afterwards to get the latest updates.

date	topic & slides	reading	assignment
Jan 13	Syllabus & Course overview	Syllabus	
Jan 15 - Jan 29	Intro to Objective-C & Foundation Kit	Programming with Obj-C	
Feb 3	Project guidelines		
Feb 5 - Feb 10	iOS SDK Overview		
Feb 12	Elevator Pitches		
	TBA		

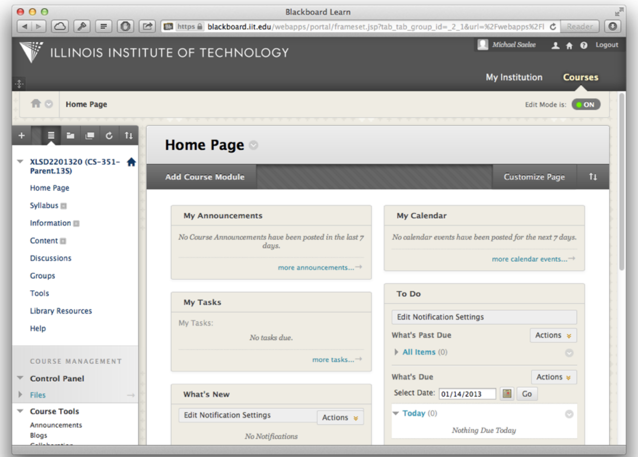
I use the abbreviations **CS:APP** for *Computer Systems: A Programmer's Perspective*, by Bryant and O'Hallaron, **K&R** for *The C Programming Language*, by Kernighan and Ritchie, and **AUP** for *Advanced Unix Programming*, by Marc Rochkind.

In addition to the readings prescribed above, there are also a number of helpful references listed in the Resources section, below.

Online resources

2. Blackboard

- *only for grade reporting*!
- feedback will be returned via a separate mechanism



No Course Announcements have been posted in the last 7 days.

No calendar events have been posted for the next 7 days.

My Tasks:

No tasks due.

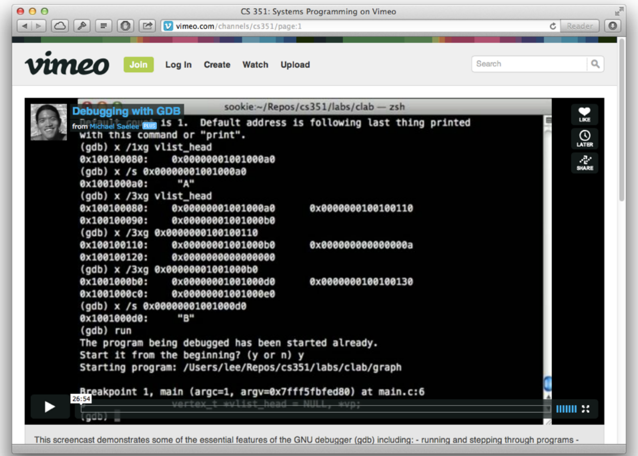
Select Date: 01/14/2013

Nothing Due Today

No Notifications

Online resources

3. Vimeo channel: screencasts
   - vimeo.com/channels/cs442
   - walkthroughs & tutorials



```
Default count is 1.  Default address is following last thing printed
with this command or "print".
(gdb) x /1xg vlist_head
0x100100080:    0x00000001001000a0
(gdb) x /s 0x00000001001000a0
0x1001000a0:     "A"
(gdb) x /3xg vlist_head
0x100100080:    0x00000001001000a0      0x0000000100100110
0x100100090:    0x00000001001000b0
(gdb) x /3xg 0x0000000100100110
0x100100110:    0x00000001001000b0      0x000000000000000a
0x100100120:    0x0000000000000000
(gdb) x /3xg 0x00000001001000b0
0x1001000b0:    0x00000001001000d0      0x0000000100100130
0x1001000c0:    0x00000001001000e0
(gdb) x /s 0x00000001001000d0
0x1001000d0:     "B"
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /Users/lee/Repos/cs351/labs/clab/graph

Breakpoint 1, main (argc=1, argv=0x7fff5fbfed80) at main.c:6
6         vertex_t *vlist_head = NULL, *vp;
(gdb) 
```

This screencast demonstrates some of the essential features of the GNU debugger (gdb) including: - running and stepping through programs -

Textbooks

- None!
  - Plenty of slides, screencasts, sample code
- developer.apple.com/ios is a great resource

Grading

- Breakdown: 50% assignments, 50% project
- No exams!
- ~6 programming assignments

Assignment grading: checkmark system

✓+ | ✓ | ✓− | 0

$\longleftarrow$ ✓+ ✓− $\longrightarrow$

Pluses/ Minuses	$+4$	$+3$	$+2$	$+1$	± 0	-1	-2	-3	-4
Letter Grade	A+	A	A-	B+	B	B-	C+	C	C-
Score (%)	100	95	92	88	85	82	78	75	72

$$0 = 3 \times (\checkmark -)$$

✓+ =

- no warnings, bugs, or crashes
- good coding style & organization
- “suggested extras” in most assignments

Project = substantial iOS app

- solo or pair work (ideally in pairs!)
- initial proposal deadline: **February 1st**
- deliverables scattered across semester

No:

- iMessage clone
- XXX reference (e.g., Matlab reference)
- 100% static / read-only apps

project deliverables:

- elevator pitch
- requirements analysis
- paper prototype / mockup
- intermediate on-device prototype
- final demo & presentation

§Class & Topics Overview

iOS Development

IIT College of Science
ILLINOIS INSTITUTE OF TECHNOLOGY

required:

- Intel Mac (Hackintosh?) & OS X 10.9+
    - iMacs in SB 108 (being updated!)
- iOS developer account for on-device testing
    - free university program invites coming
    - $99 for App store deployment

Xcode   *iOS Simulator*   *Instruments*

Development Tools

language: Objective-C

- OO, dynamically typed superset of C
  - open source runtime and compiler
- Fairly small language specification

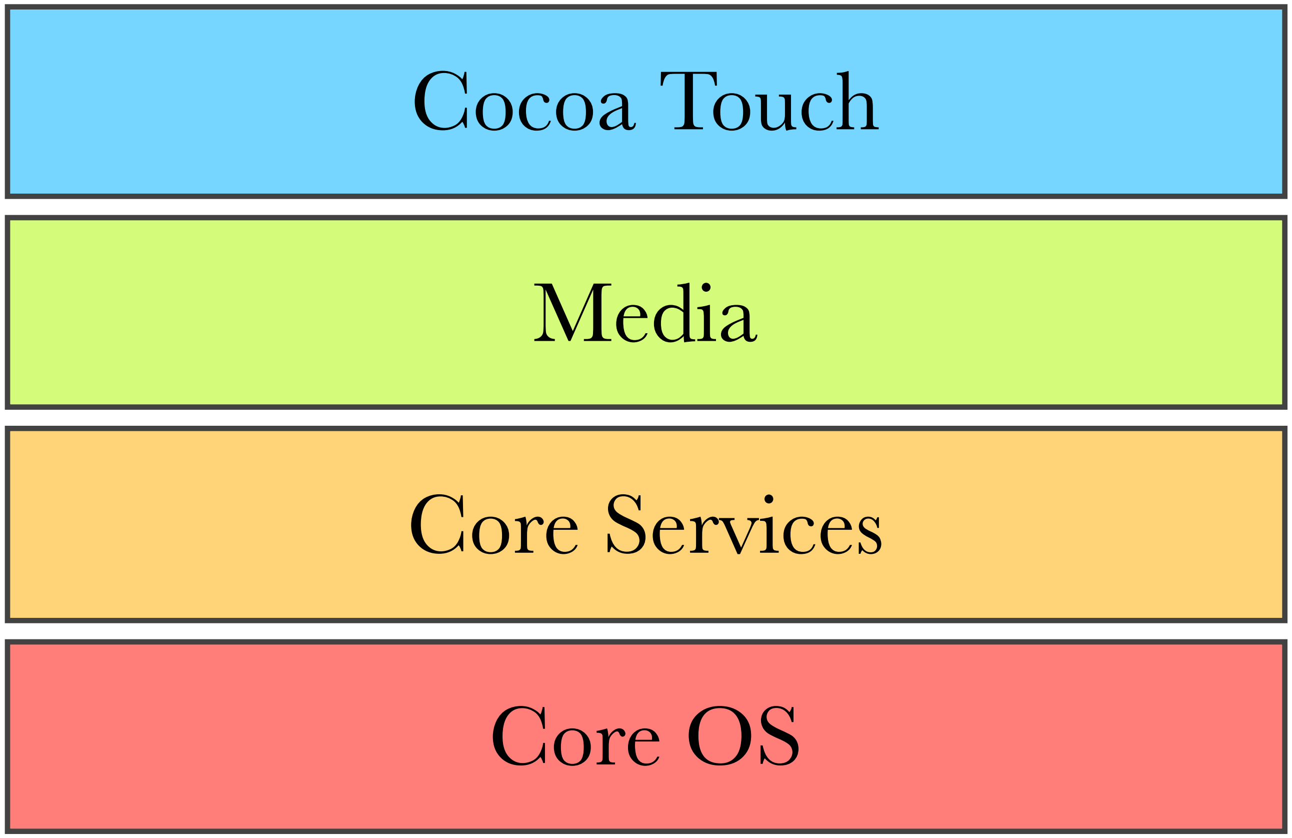
new language: Swift!

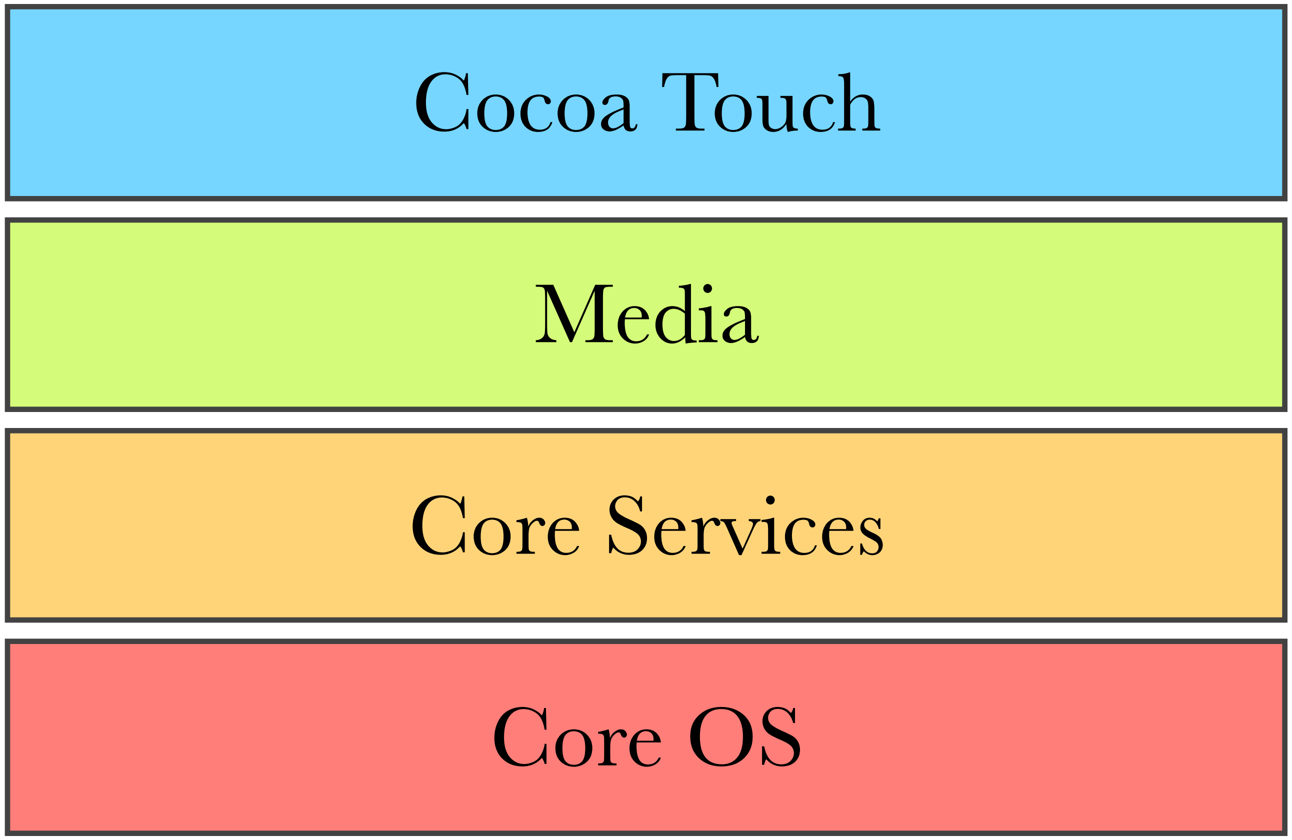
- multi-paradigm, strongly typed, first-class functions, type inference (& more)
- shares runtime with ObjC platform
    - fully cross-compatible (language/libraries)

iOS API inherits a lot from the OS X platform

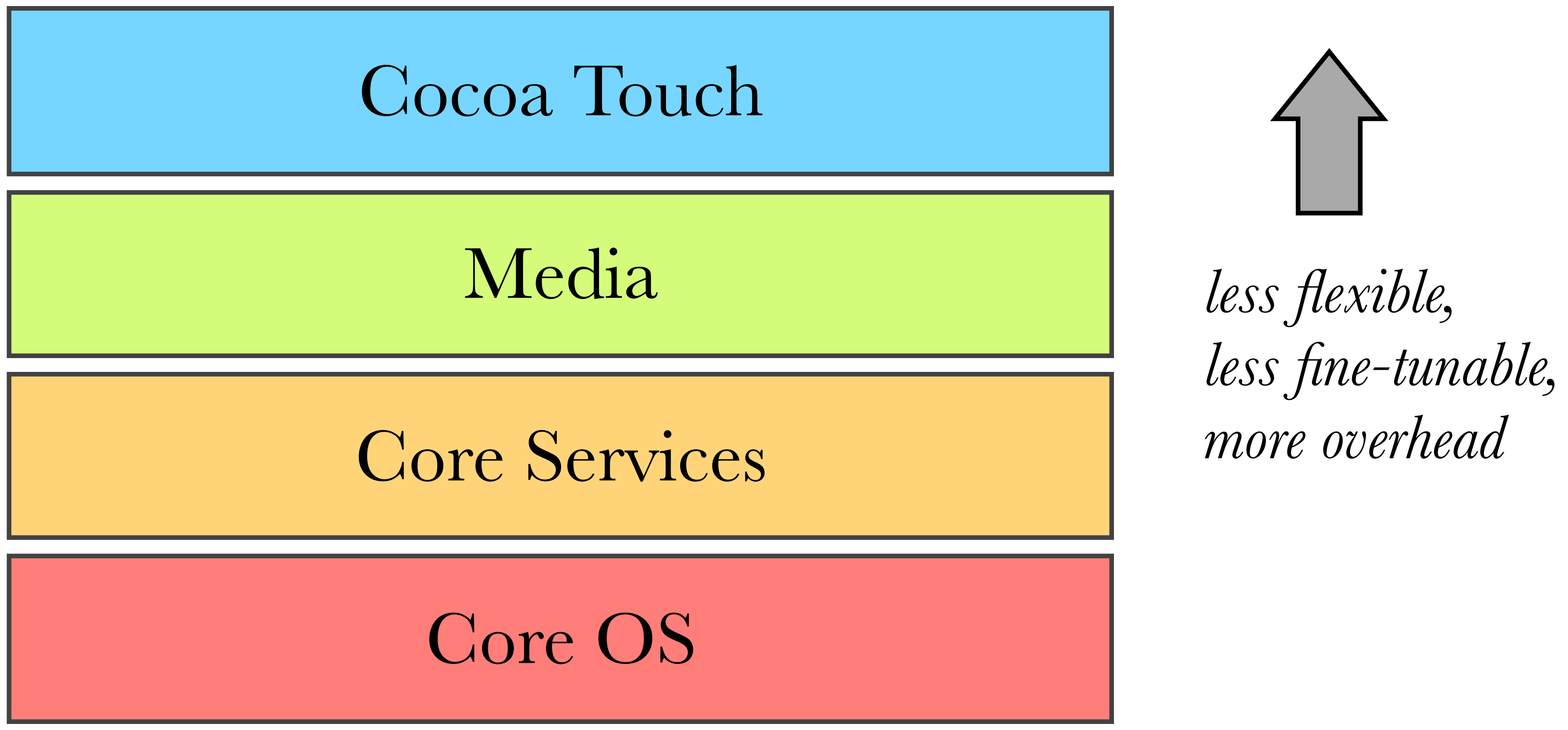
- NeXTSTEP ancestry
- Low level APIs (e.g., data structures, threading, networking)
- “Core” APIs: graphics, animation, etc.

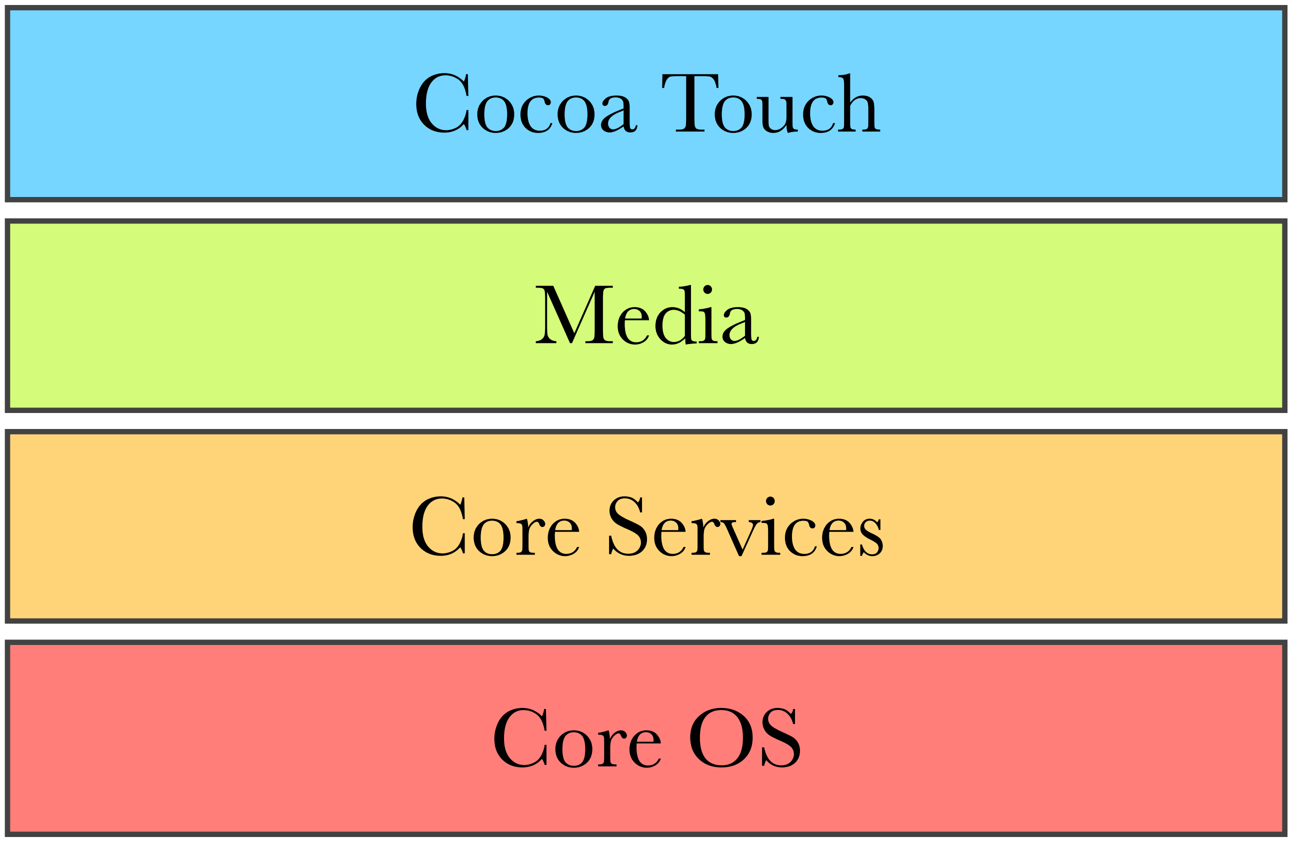
Apple-provided frameworks fall into different layers of the iOS *architectural stack*



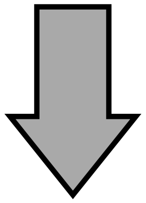


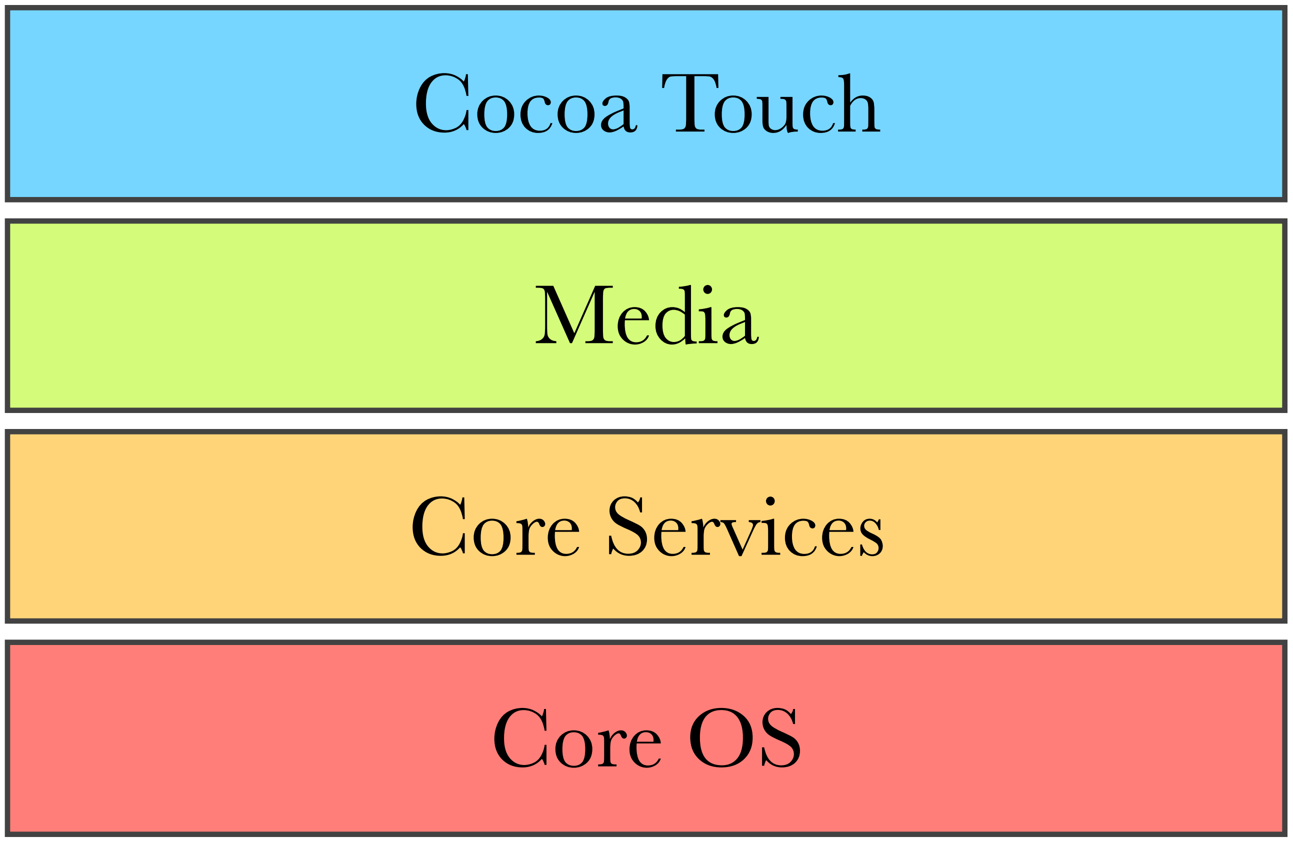
object-oriented,
more abstract,
less code needed


Cocoa Touch
Media
Core Services
Core OS
less flexible,
less fine-tunable,
more overhead

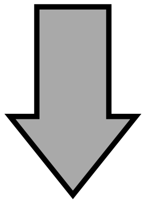


may be procedural, more granular, exposes hardware





complex APIs,
more details …
… more code!

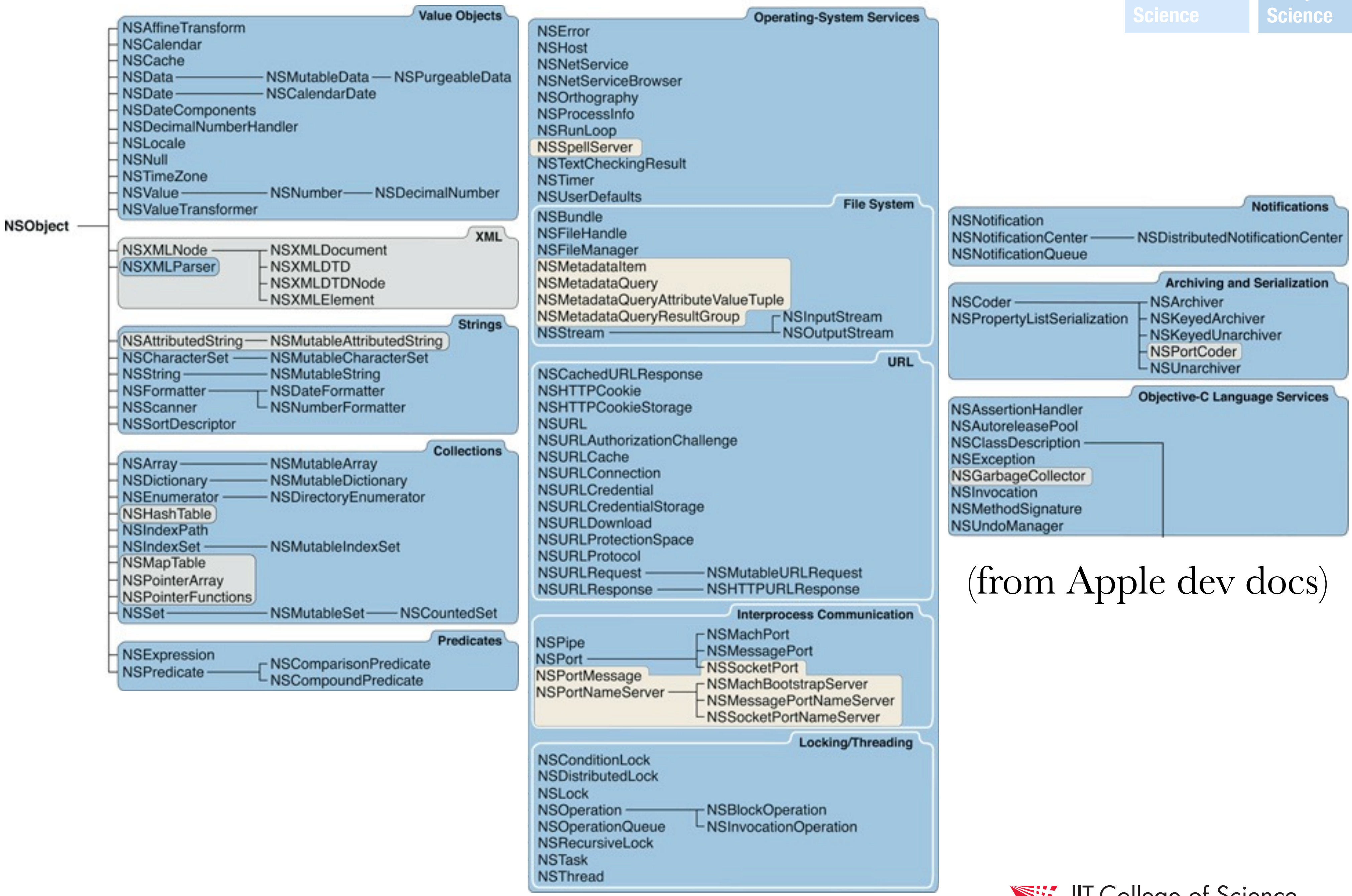


Core OS

- Unlikely to use directly, but used by other layers of iOS stack
- e.g., Security, Bluetooth and System APIs (POSIX / Unix)

Core Services

- "Core" system services for all iOS apps
- Infrastructure: iCloud, In-App Purchase, Newsstand, Social, etc.
- Hardware: Location, Motion, Telephony
- Data structures/management: Core data, **Foundation** framework



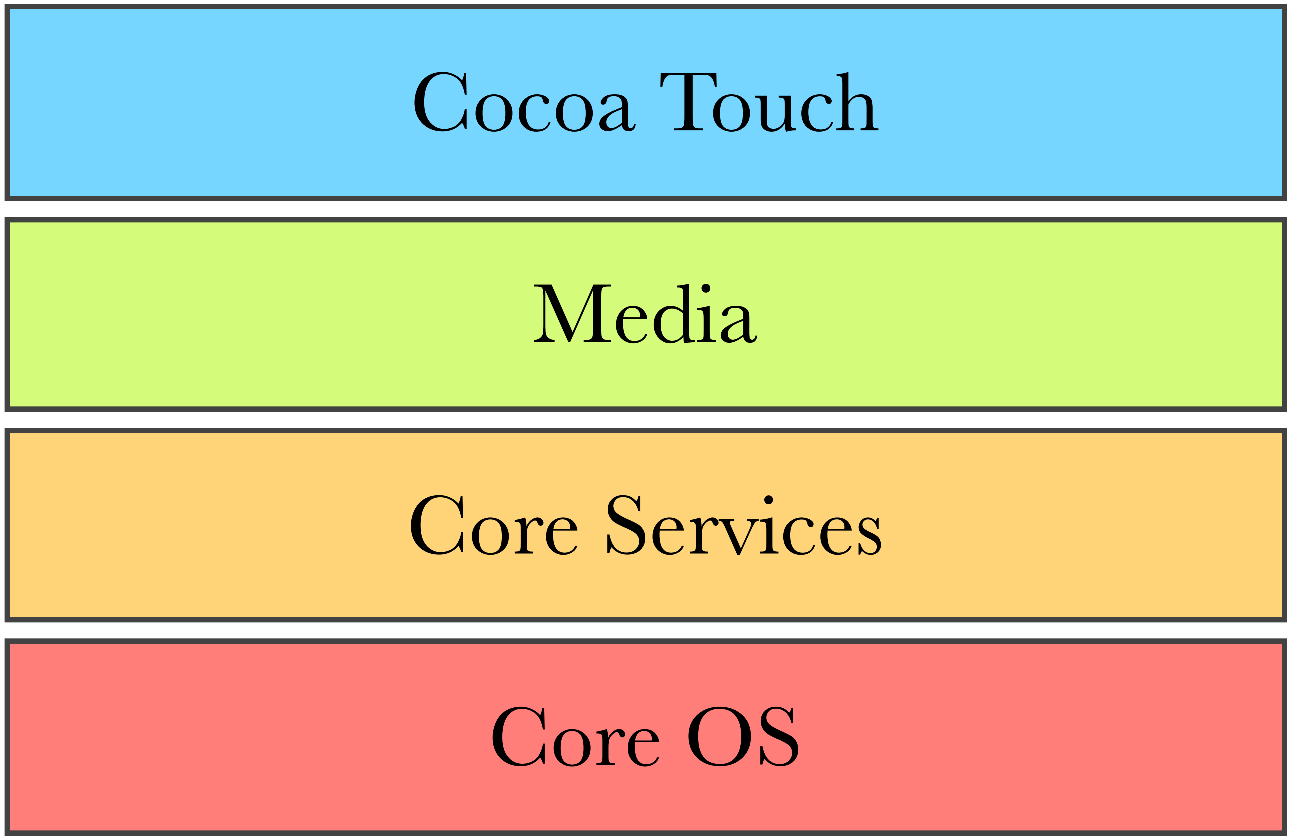
(from Apple dev docs)

Media

- Graphics, Audio, Video APIs
- Core Graphics/Animation/Image/etc.
  - e.g., custom 2D drawing and rendering
- OpenGL ES
  - hardware accelerated 2D/3D graphics

Cocoa Touch

- High level app infrastructure
  - e.g., touch-events, on-screen interface elements, transitions, gestures
- Built-in controllers (e.g., map, photopicker)
- Key framework: **UIKit**


Cocoa Touch
Media
Core Services
Core OS

Typically many ways to accomplish a given task!
(i.e., with frameworks at different levels)

UIKit

```
// clear with white rectangle
[[UIColor whiteColor] set];
UIRectFill(self.bounds);

// load and draw image at (0,0)
[[UIImage imageNamed:@"image.png"] drawAtPoint:CGPointMake(0, 0)];
```

Core Graphics

```
// get current graphics context to draw into
CGContextRef context = UIGraphicsGetCurrentContext();

// clear with white rectangle
CGContextSetRGBFillColor(context, 1.0, 1.0, 1.0, 1.0);
CGContextFillRect(context, self.bounds);

// load image from file
NSString* imageFileName = [[[NSBundle mainBundle] resourcePath]
                           stringByAppendingPathComponent:@"image.png"];
CGDataProviderRef provider = CGDataProviderCreateWithFilename([imageFileName UTF8String]);
CGImageRef image = CGImageCreateWithPNGDataProvider(provider,
                                                    NULL,
                                                    true,
                                                    kCGRenderingIntentDefault);
CGDataProviderRelease(provider);

// draw image at (0,0)
CGContextDrawImage(context,
                   CGRectMake(0, 0, CGImageGetWidth(image), CGImageGetHeight(image)),
                   image);
CGImageRelease(image);
```

academic value?

- not just APIs
- focus on design techniques & best practices

broader concerns:

- software design patterns
- testing (functionality, performance)
- prototyping workflow
- version control

Coming up: Swift