Lecture 2

CS2049: Intermediate iPhone Development

Instructor: Daniel Hauagge

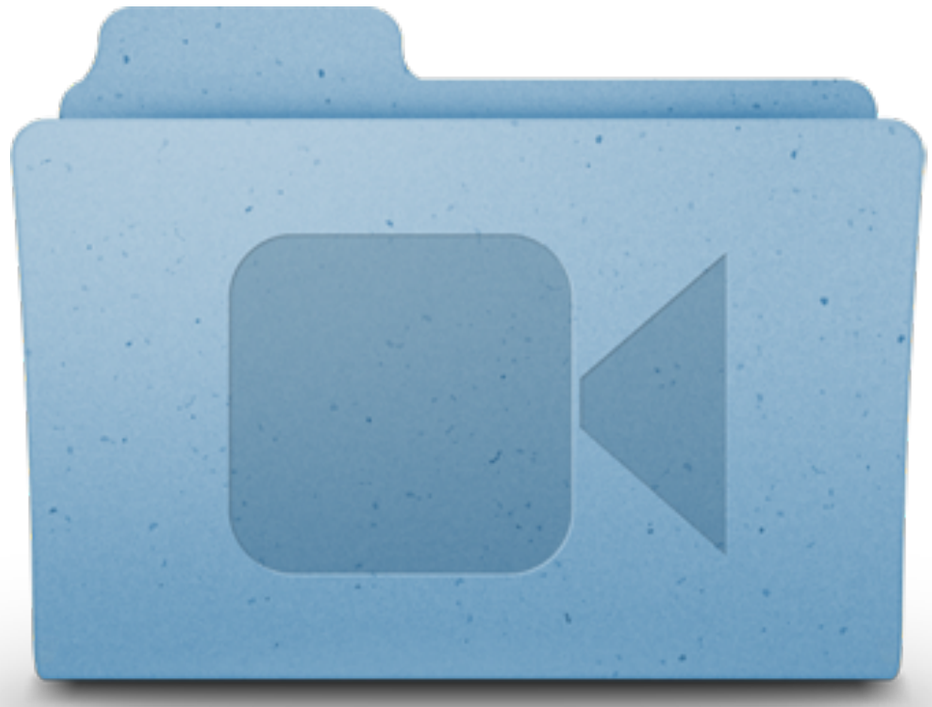
Today's Lecture

- Frameworks
  - AVFoundation
  - CIDetector
    - Layers and Views
- Language features:
  - Lazy initialization
  - try!

Today's App

Video Capture

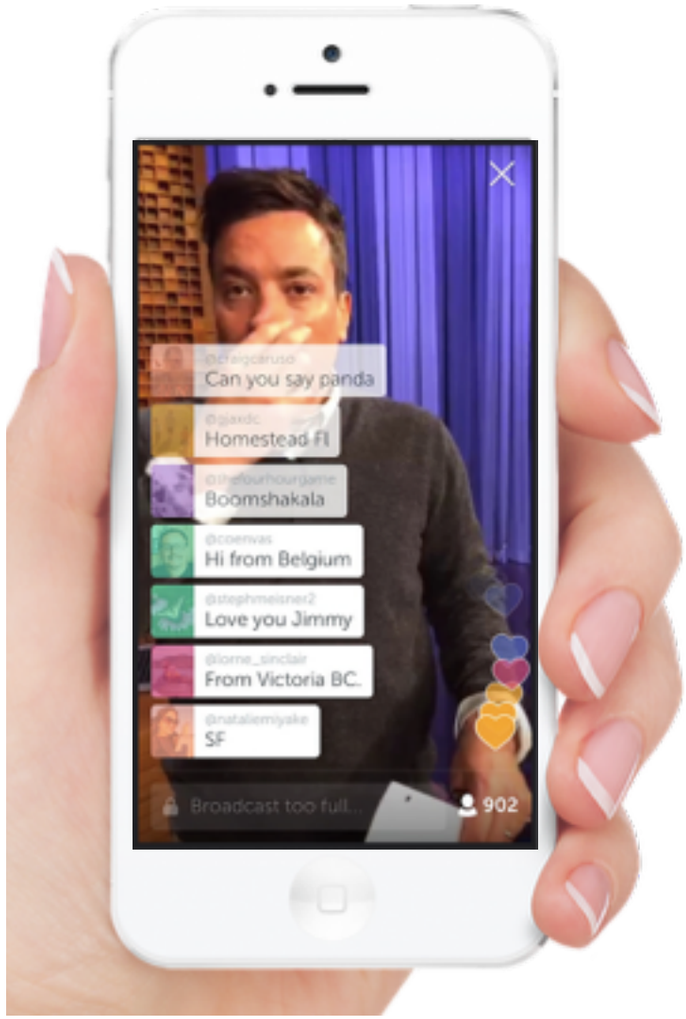
with AVFoundation

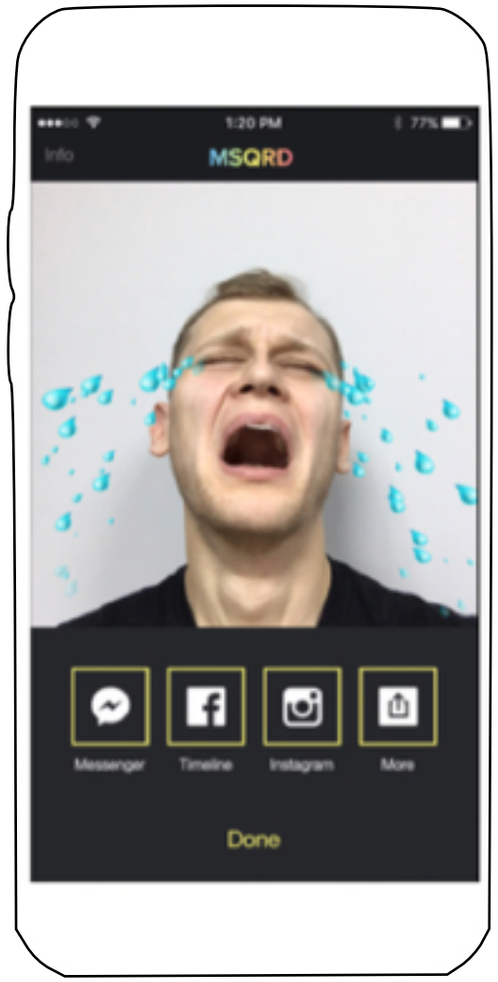


iPhone Camera

Front Camera	Rear Camera
5MP	12MP
HD video	4K video
Burst Mode	Burst Mode
Retina Flash	True Tone Flash
	Slow motion: 120 to 240 FPS
	Video stabilization (optical and digital)
	Time lapse video
	Panoramas

Cool Apps





Instagram HYPERLAPSE

periscope

MSQRD

Levels of

UIImagePickerController	AVFoundation
Easy of use	Greater control
Only one class to deal with	Breaks down pipeline into input, output, session management
Provides UI for capture	

AVFoundation

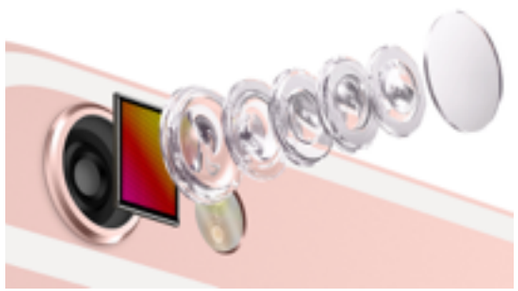
- Still and video capture
- Video preview using CALayer
- Audio level metering
- Access to raw data
- Video stabilization
- HDR Video

AVFoundation

- Makes sense when:
  - You need fine grained control
  - Independent points of interest for focus or exposure
  - Access to video frame data (with accurate timing data)
  - Per frame metadata (e.g., exposure)
  - Configurable frame-rate, output format, resolution

Capture Basics

Hardware



Inputs

- Camera
- Microphone

Output

- Compressed Video to File
- Still Frame
- Audio Data
- Video Data

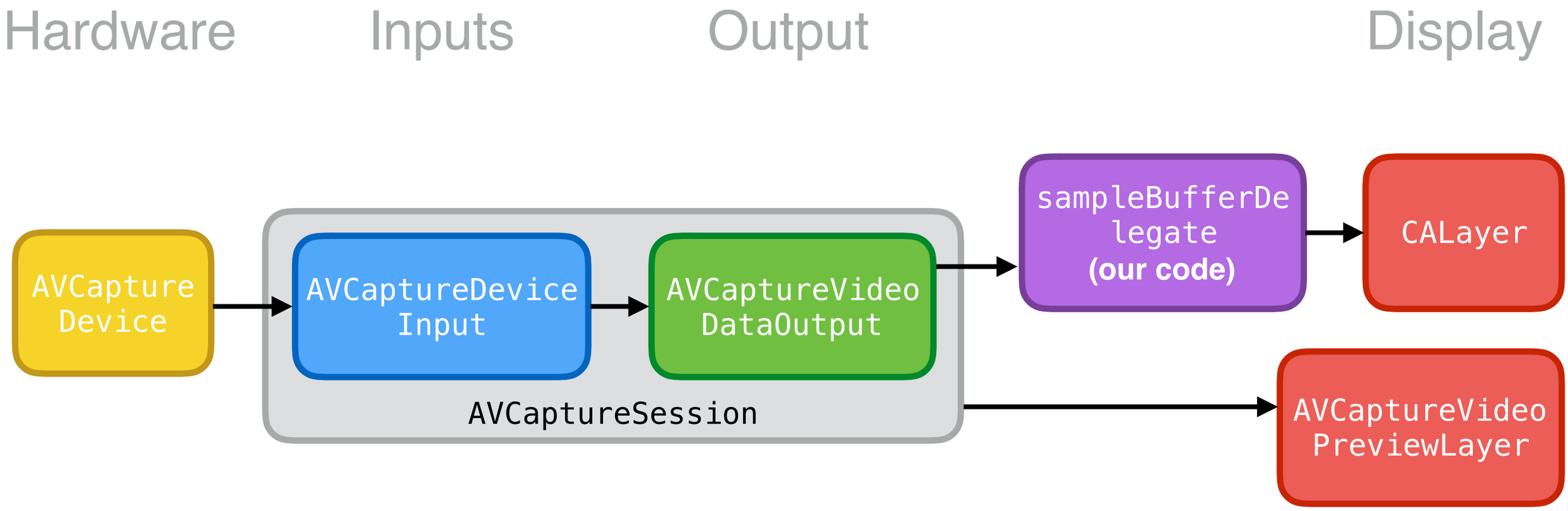
Capture Basics

Hardware Inputs Output Display

Open this webpage to see instructors code

http://10.148.6.140:8000

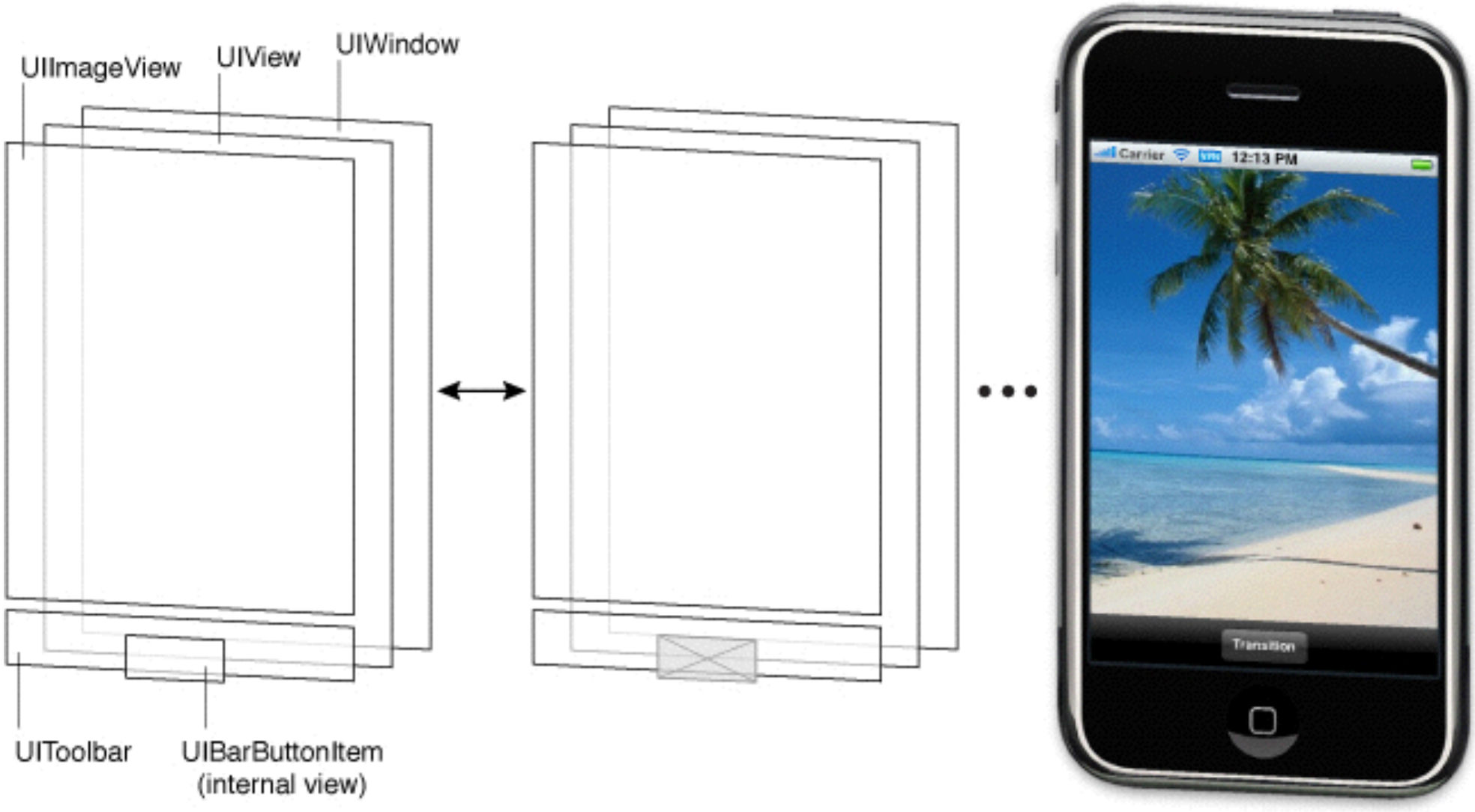
Capture Basics



Layers and Views

UIView	CALayer
Can receive touch events	Represent position, shape and anchor point
Are always backed by a Layer on iOS	Do not receive touch events Light-weight
No implicit animations	Implicit animations

Layers and Views



CIDetector

- Faces
  - Eyes
  - Smile detection
  - Face angle
- Rectangles
- QRCodes
- Text

try!

```
var x = try! someFunc()
```

≈

```
do {
  var x = someFunc()
} catch {
  assert(false)
}
```

- In case you are sure an exception wont be thrown
- Compiler wraps code in runtime assertion
- Runtime error (you app dies) if an actual exception is thrown

try?

```
var x = try? someFunc()
```

≈

```
var x : Int?
do {
  x = someFunc()
} catch {
  x = nil
}
```

- An exception can be thrown but you can continue by setting the variable to nil
- No runtime error

guard

```
var y : Int?

guard var x = y else {
  return
}
```

≈

```
var y : Int?

if y == nil {
  return
}

x = y!
```

- An exception can be thrown but you can continue by setting the variable to nil
- No runtime error

guard