Animations & Sound

PROG31975 – Week 3 Part 1

Jawaad Sheikh

Jawaad.Sheikh@SheridanCollege.ca

Outline

- iOS Sound
  - AVFoundation
  - Audio Playback
  - OpenAL
- Core Animation
  - ImageViews
  - Animating Your Graphics
- Exercises

iOS Sound

- The iPhone platform has extensive multimedia capabilities.
- It can play both audio and video.
- It can record both audio and video.

iOS Sound

- The platform also provides the framework to developers to create their own apps to perform these capabilities.
- This class will focus on audio playback.

iOS Sound

- For the developer, there are a number of different ways to implement audio.
  - Media player framework.
  - AV Foundation framework
  - Audio Toolbox framework
  - Audio Unit framework
  - OpenAL framework

iOS Sound

- Media Player framework
  - Used to play songs, audio books and podcasts from the iPod library
- AV Foundation framework
  - Used to play and record audio using a simple Objective-C / Swift interface.

iOS Sound

- Audio Toolbox framework
  - Used to play audio with synchronization capabilities, access packets of incoming audio, parse audio streams, convert audio formats, etc.

iOS Sound

- Audio Unit framework
  - Used for connecting and using audio processing plugins.
- OpenAL framework
  - To provide audio playback.
  - iOS supports OpenAL 1.1

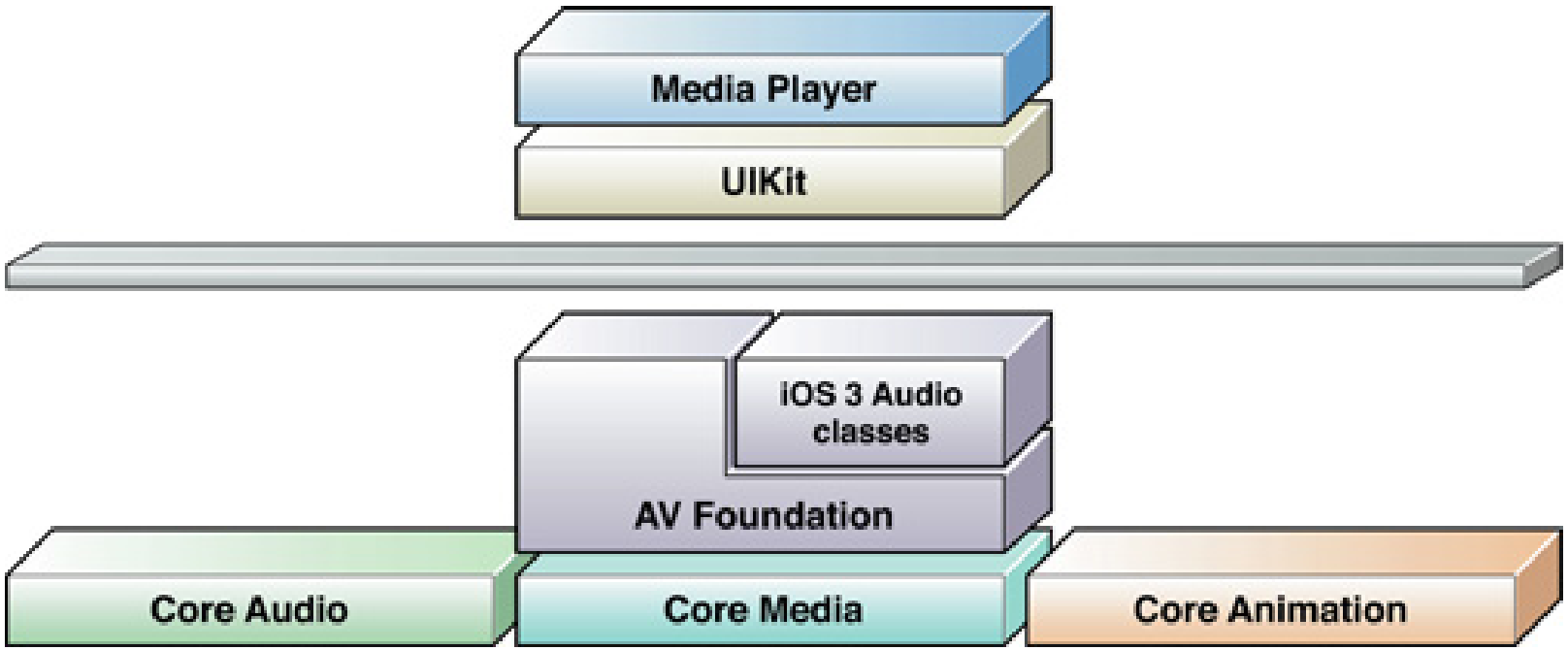
iOS Sound

- We will focus on:
  - AV Foundation
  - OpenAL

AV Foundation

- This framework plays both audio and video files.
- There are two main frameworks:
  - Core Audio
  - Core Media

AV Foundation



AV Foundation

- To work with video, you would use the following classes:
  - MPMoviePlayerController
  - MPMoviePlayerViewController

AV Foundation

- To work with audio, you would use the following classes:
  - AVAudioPlayer
  - AVAudioRecorder

AV Foundation

- Callouts from AVFoundation are not guaranteed to be made.
- Because these calls are made on threads, you are responsible for ensuring that your callout is invoked on the correct thread.

AV Foundation

- If you’re using threads, you can take advantage of NSThread’s “isMainThread” method to ensure you’re on the right thread.

AV Foundation

- There are two approaches to connecting with an audio (or video for that matter) file.
  - Providing a (file) URL
  - Using an Asset

Audio Playback

- Playing back audio is very easy.
- In your view controller’s class def you’ll need to implement AVAudioSessionDelegate and AVAudioPlayerDelegate

Audio Playback

- Inside viewDidLoad (or appear) (assumes file url):

```
// assume var soundPlayer : AVAudioPlayer?
let soundURL = Bundle.main.path(forResource:
  “songname”, ofType: “mp3”)
let url = URL(fileURLWithPath: soundURL!)
soundPlayer = try! AVAudioPlayer.init(contentsOf: url)
soundPlayer?.currentTime=0
soundPlayer?.volume = 0.5
soundPlayer?.numberOfLoops = -1
soundplayer?.play()
```

Audio Playback

- The class, AVAudioSession, that is instantiated is a class needed to:
  - Activate or deactivate your app’s audio session
  - Set the audio session category and mode
  - Specify your preferred audio hardware sample rate and I/O buffer duration

Audio Playback

- The class, AVAudioPlayer, that is instantiated is a class needed to:
  - Provide playback of the audio file.
  - Adjust and set properties of the file (volume, rate, etc.)

Audio Playback

- Where is the file located?
  - Similar idea to using images.
  - Create a folder to hold your audio files.
  - Add your audio files into your Xcode project

Audio Playback

- To play the file:

  soundPlayer.play()

- To stop the file:

  soundPlayer.stop()

Audio Playback

- Support Method:
  - audioPlayerDidFinishPlaying

OpenAL

- A cross platform 3D audio API for gaming and other applications.
- More can be found at www.openal.org
- Basic OpenAL components are:
  - Listeners
  - Sources
  - Buffers

OpenAL - Source: http://elevatedpixels.com/?p=117

- One thing that makes OpenAL different is that you need to convert your files to the “caf” format before adding it to your project.
- From a terminal (assume your file is called MainMenu.mp3)

/usr/bin/afconvert -f caff -d LEI16 /MainMenu.mp3 /MainMenu.caf

OpenAL

- Note that OpenAL is only used for sounds less than 30 seconds!
- So it is not needed for extensive background music.
- Solution is to use AVAudioPlayer for background music and use OpenAL to play sound effects.

ImageView’s

- In order to display a graphic, you need to use an ImageView.
- There is another way, which we’ll look at later.
- There are 2 classes you need to work with:
  - UIImageView
  - UIImage

ImageView’s

- UIImageView
  - The object that displays a UIImage object.
- UIImage
  - The object that holds the actual image.

ImageView's

- There's 2 approaches to getting an image to appear
  - Hard code it directly into the Storyboard – fine for static images.
  - Using UIImage – allows you to customize the image for different needs.

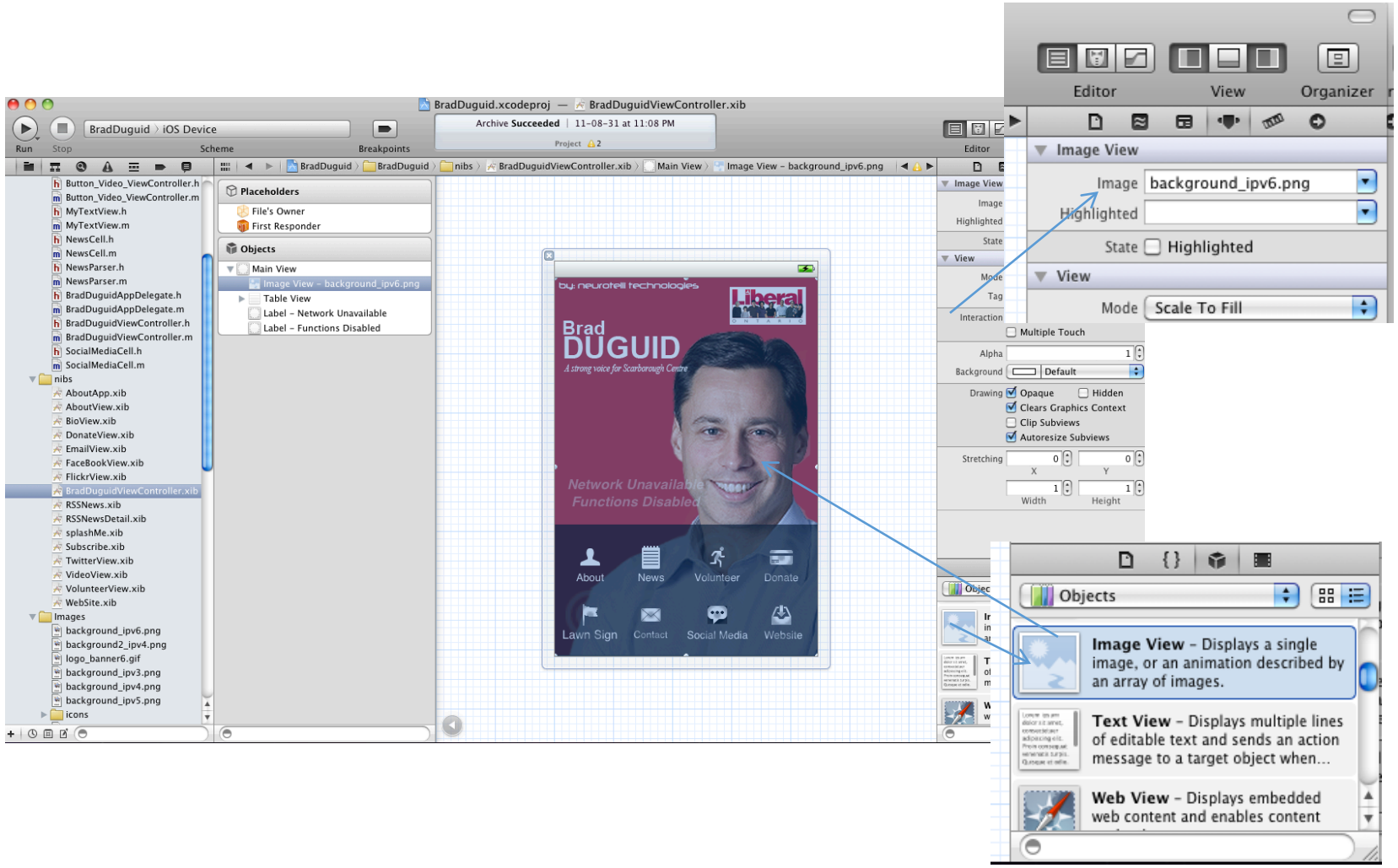
ImageView Example 1

- Static image approach
- If you haven’t already done so, in your project workspace, create a new group and call it “images”.
  - We’ll hold all our images here.
- Add a new file to your images folder.
  - You’re adding a new image you want to display.

ImageView Example 1

- In your Storyboard drag a UIImageView object onto your view.
- In the Attributes Inspector, select your new image file under “image”

ImageView Example 1



ImageView Example 2

- Coded image approach.
- Define your UIImageView object

  @IBOutlet var myImageView:UIImageView!
- Somewhere you can define your image as:

  let myImg = UIImage(named: “filename.jpg”)

  myImageView.image = myImg

ImageView Example 3

- A third approach is to have your ImageView become a slide show.
- This approach involves creating an array of image file names and setting up a timer for when to change the image.

ImageView Example 3

- Creating an animating image involves using an array of images

```
// assume UIImageView is called imgView
let img1 = UIImage(named:”firstImage.jpg”)
let img2 = UIImage(named:”secondImage.jpg”)
let arImg = [img1,img2]

imgView.animationImages = arImg
imgView.animationDuration = 2.0  // 1 cycle lasts 2 seconds.
imgView.animationRepeatCount = 0 // repeats forever
imgView.startAnimating()
```

Animating your Graphics

- iOS has a framework that helps you handle animating images.
- The framework is called “Quartz2d”
- You need to add this framework in order to make this work.

Animating your Graphics

- CoreGraphics.framework
  - Quartz 2D API manages the graphic context and implements drawing.
  - Quartz Services API provides low level access to the window server. This includes display hardware, resolution, refresh rate, and others.
- QuartzCore.framework
  - Core Animation: Objective-C/Swift API to do 2D animation.
  - Core Image: image and video processing (filters, warp, transitions).iOS 5

Animating your Graphics

- You have the ability to:
  - Translate an image across the screen
  - Shrink / grow an image
  - Rotate an image
  - A combination of these
  - And more.

Animating your Graphics

- This example will focus on translating an image across the screen.
- Core Animation works in “layers”.
- Each layer is of type “CALayer”
- A layer can be animated using “CABasicAnimation”

ImageViews Example 4 (Using CALayer)

- This example displays an image **without** UIImageView
- Instead, we use CALayer.

```
//assume var spinLayer : CALayer?
let spinImage = UIImage(named: “file.jpeg”)
spinLayer = CALayer.init()
spinLayer?.contents = spinImage.cgImage
spinLayer?.bounds = … // size
spinLayer?.position = … // location
view.layer.addSublayer(spinLayer!)
```

Animating your Graphics

- Both classes together, can get your layer to do anything you want.
- In the View Controller in which you want to animate an image add the following to viewDidLoad: (assume your file is called “card.png”)

Animating your Graphics

```
// This code handles moving the image
// assume var flyLayer : CALayer?
let moveAnimation = CABasicAnimation(keyPath: ”position”)
moveAnimation.timingFunction =
CAMediaTimingFunction(name:kCAMediaTimingFunctionEaseInEaseOut)
moveAnimation.fromValue = NSValue.init(cgPoint:CGPoint(x:0, y:0))
moveAnimation.toValue = NSValue.init(cgPoint:CGPoint(x:700, y:500))
moveAnimation.duration = 3.0
moveAnimation.repeatCount = Float.infinity
flyLayer?.add(moveAnimation, forKey:nil)
```

Animating your Graphics

// to rotate your image 360 degrees

```
// assume var spinLayer : CALayer?
let rotateAnimation = CABasicAnimation(keyPath:”transform.rotation”)
rotateAnimation.timingFunction =
CAMediaTimingFunction(name:kCAMediaTimingFunctionEaseInEaseOut)
rotateAnimation.fromValue = 0
rotateAnimation.toValue = 2 * Double.pi
rotateAnimation.duration = 1.0
rotateAnimation.repeatCount = Float.infinity
spinLayer?.add(rotateAnimation, forKey:nil)
```

Exercise 1

- Create an iPhone app with a home page and 3 sub pages.
- On all 3 sub pages have a slider to control volume and play a different song using AVAudioPlayer
- On sub page 1 have an image rotate
- On sub page 2 have an image move across the screen
- On sub page 3 have an image fade out

Exercise 2

- Create an iPhone app with a home page and 2 sub pages
- On sub page 1, have an image displayed using CALayer with a segmented control that will allow the user to rotate, fade or move that image on the screen.
- On sub page 2 have a slider for volume control and a segmented control such that each segment plays a different song and the slider adjust the volume of the song selected.