ATLS 4120/5120: Mobile Application Development
Week 8: Audio

We're going to build an app that records and plays back audio.

Create a new Single view application named audio and this time make it Universal.
Click on the Target and go into the Build Phases tab.
Open Link Binary with Libraries and click the + to add the AVFoundation framework under iOS 8.4.

In the storyboard add 3 buttons for Record, Play, and Stop. (centered)
Connect these as outlets called recordButton, playButton, and stopButton. We need these so we can enable and disable these buttons as needed.
Also connect these as actions called recordAudio, playAudio, and stopAudio.
Resolve auto layout issues for all views with Reset to Suggested Constraints.

For audio we need the AVFoundation framework for audio so add that to your project.
We also need to adopt the AVAudioPlayerDelegate and AVAudioRecorderDelegate

ViewController.swift

```
import AVFoundation

class ViewController: UIViewController, AVAudioPlayerDelegate,
AVAudioRecorderDelegate
```

Create an instance variable for our audioplayer and audiorecorder

```
    var audioPlayer: AVAudioPlayer?
    var audioRecorder: AVAudioRecorder?
```

Create a constant for the name of the file where the audio will be saved

```
      let fileName = "audio.caf"
```

viewDidLoad is a good place to do our setup and initialization

```
    override func viewDidLoad() {
        //disable buttons since no audio has been recorded
        playButton.enabled = false;
        stopButton.enabled = false;

        //get path for the audio file
        let dirPath =
NSSearchPathForDirectoriesInDomains(NSSearchPathDirectory.DocumentDirectory,
NSSearchPathDomainMask.UserDomainMask, true)
        let docDir = dirPath[0] as! String //documents directory
        let audioFilePath = docDir.stringByAppendingPathComponent(fileName)
        let audioFileURL = NSURL(fileURLWithPath: audioFilePath) //URL to
the audio file

        //recorder settings
        //NSDictionary for settings
        let recordSettings = [AVEncoderAudioQualityKey:
AVAudioQuality.Min.rawValue, AVEncoderBitRateKey: 16, AVNumberOfChannelsKey:
2, AVSampleRateKey: 44100.0]
```

```
        var error : NSError?

        //create the AVAudioRecorder instance
        audioRecorder = AVAudioRecorder(URL: audioFileURL, settings: 
recordSettings as [NSObject : AnyObject], error: &error)

        //test for error
        if let err = error {
            println("AVAudioRecorder error: \(err.localizedDescription)")
        } else { //no error
            audioRecorder?.delegate = self //sets the delegate
            audioRecorder?.prepareToRecord() //ready to record
        }
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a 
nib.
    }
```

Now implement the 3 methods

```
    @IBAction func recordAudio(sender: UIButton) {
        //if not already recording, start recording
        if audioRecorder?.recording == false{
            playButton.enabled = false
            stopButton.enabled = true
            audioRecorder?.record()
        }
    }

    @IBAction func playAudio(sender: UIButton) {
        //if not recording play audio file
        if audioRecorder?.recording == false{
            stopButton.enabled = true
            recordButton.enabled = false
            var error: NSError?

            //create the AVAudioPlayer instance
            audioPlayer=AVAudioPlayer(contentsOfURL: audioRecorder?.url, 
error: &error)

            //test for error
            if let err = error {
                println("AVAudioPlayer error: \(err.localizedDescription)")
            } else {
                audioPlayer?.delegate=self //sets the delegate
                audioPlayer?.play() //plays audio file
            }
        }
    }

    @IBAction func stopAudio(sender: UIButton) {
        stopButton.enabled = false
        playButton.enabled = true
        recordButton.enabled = true
        //stop recording or playing
        if audioRecorder?.recording == true {
            audioRecorder?.stop()
        } else {
            audioPlayer?.stop()
        }
    }
```

Both delegate protocols have optional methods. Since you don't press a button when the audio playing ends, let's implement that one to change the buttons as needed.

```
//AVAudioPlayerDelegate method

    func audioPlayerDidFinishPlaying(player: AVAudioPlayer!, successfully
flag: Bool) {
        recordButton.enabled = true
        stopButton.enabled = false
    }
```

App icons

Since we created this as Universal if you look at the App icons you will see there are spots for iPhone and iPad. You need iPhone and iPad App icons, 4 in total.