Nick Keirstead

5/13/15

Lesson 1

1. **What do you think this code does?** I believe the code defines an AVAudioPlayer and stores it as an implicitly unwrapped optional in “audioPlayer” variable. It then takes some sort of file and checks whether its filePath equals “love\_story” and checks if it is an mp3 file. If so, it sets the audioPlayer to be an AVAudioPlayer that plays the contents of the file’s url, and sets the error-state as nil (no error). If not, it prints that the file is empty. At the end, it calls the audioPlayer’s play method, which will presumably play the mp3 file if the correct one was loaded, or play nothing if it wasn’t and the error state was not nil. If I had to guess, this code belongs either in the iTunes app, or in the web browser’s mp3 player that can play songs off websites.
2. **8 Things I Would Need To Figure Out How To Do:** (5-8 for final version)
   * + 1. Add Buttons and customize their appearance
       2. Add audio files
       3. Link buttons to audio files
       4. Transition from startup screen to button-choice screen
       5. Receive user’s voice input through microphone
       6. Convert microphone input to sound file
       7. Run a sound file through a filter and distort it
       8. Link usage of a filter to a button
3. **What would you name the different XCode sections?**

Left: Document Navigtor

Center: Editor

Right: Reference

**4.  Explore XCode**

1. **How would you use XCode to run the current version of the app in a simulator?** I would press the play arrow in the upper left corner of XCode’s window to run the app in OSX’s built-in ‘IOS Simulator’
2. **How would you check the CPU and memory usage for the current version of the app?** While running the simulator, click on the third button from the right end of the navigator with popup text “Show the debug navigator.” In the debug navigator, CPU and memory statistics can be found.

Lesson 2

1. **What would you do to center align this button both vertically and horizontally?** Control click and drag the button to the top edge of the screen and release, then select center horizontally in container. Do the same to the right or left edge of the screen and select center vertically in container.
2. **What does the circle right next to the action in the gutter indicate?** Upon clicking on the circle (next to a given IBAction), a popup menu shows the objects (buttons, labels, etc.) in the view connected to the piece of code.
3. **What information does the Connection Inspector provide?** The Connections Inspector details the outlets and sent events each object in the view is tied to, such as what actions may be invoked when the user chooses to “Touch Up Inside”, etc.
4. **What comes to mind when I say the word outlet?** A place where you plug something in to make a connection.
5. **Is ViewController.swift a View or a Controller?** It is a controller, as it contains code that governs the behavior of objects in the view, which is “Main.storyboard”
6. **What are the differences between an action and an outlet?** An action is the mechanism by which the view can “communicate” with the controller — in many cases, it is a function that is called that handles certain onscreen events that have triggered the action in the first place. It is a blind form of communication going one way. An outlet is the way in which the controller can govern aspects of the view, such as a variable that defines a button that will appear.