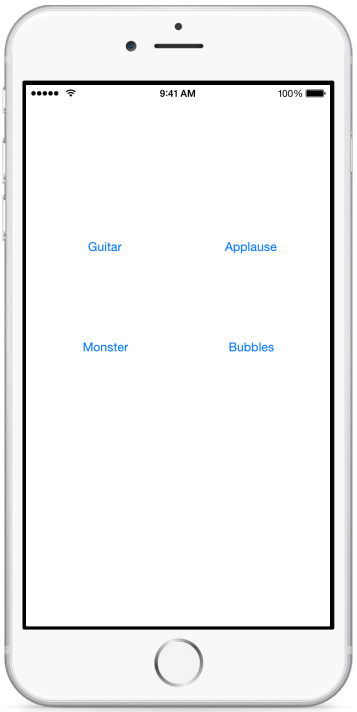
NoiseMaker
Lesson 4

Description

Add four independent `AVAudioPlayer?` properties to the controller, and invoke each player in a respective controller action.



Learning Outcomes

- Practice declaring properties and implementing controller methods.
- Discover the concept of asynchronous methods, and relate asynchronous method calls to how execution can continue while a sound is playing.

Vocabulary

property	`AVAudioPlayer`	optional
URL	`NSURL`	app bundle
`NSBundle`	asynchronous method	

Materials

- **NoiseMaker Lesson 4** Xcode project
- **Asynchronous Methods** presentation

Opening

How do we get each sound to play independently, so one sound doesn't stop when another begins playing?

Agenda

- In the `ViewController` class, replace the single controller `player` property with four distinct `AVAudioPlayer?` properties.

```
var guitarPlayer: AVAudioPlayer?
var applausePlayer: AVAudioPlayer?
var monsterPlayer: AVAudioPlayer?
var bubblesPlayer: AVAudioPlayer?
```

- Discuss how each property is an optional, since the `ViewController` initializer will not initialize the properties with values.
- Update the implementation of each controller action to use each relevant `AVAudioPlayer?` property.

```
@IBAction func playGuitar(sender: UIButton) {
   if let url = NSBundle.mainBundle().URLForResource("guitar",
      withExtension: "wav") {
      guitarPlayer = try? AVAudioPlayer(contentsOfURL: url)
      guitarPlayer?.play()
   }
}
...
@IBAction func playBubbles(sender: UIButton) {
   if let url = NSBundle.mainBundle().URLForResource("bubbles",
      withExtension: "wav") {
      bubblesPlayer = try? AVAudioPlayer(contentsOfURL: url)
      bubblesPlayer?.play()
   }
}
```

- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `AVAudioPlayer play` method, and notice the description "play a sound asynchronously."
- Present the concept of asynchronous execution.
- Run the app (⌘R), tap each button in quick succession, and listen to multiple sounds playing simultaneously.

Closing

What do you think about the repetitive code in our controller? Where is the model, and what do you think it should it be?

Modifications and Extensions

- Explore the `AVAudioPlayer` API and experiment with additional methods for manipulating audio playback. Change the playback volume and stereo balance, and use the `numberOfLoops` property to cause the sounds to repeat.

Resources

Bundle Programming Guide: Accessing a Bundle's Contents http://developer.apple.com/library/ios/documentation/CoreFoundation/Conceptual/CFBundles/AccessingaBundlesContents/AccessingaBundlesContents.html

NSBundle Class Reference https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/NSBundle_Class/index.html

NSURL Class Reference https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/NSURL_Class/index.html

AVAudioPlayer Class Reference https://developer.apple.com/library/ios/documentation/AVFoundation/Reference/AVAudioPlayerClassReference/index.html

The Swift Programming Language: If Statements and Optional Binding https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/TheBasics.html#//apple_ref/doc/uid/TP40014097-CH5-ID333

The Swift Programming Language: Error Handling https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/ErrorHandling.html

The Swift Programming Language: Optional Chaining https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/OptionalChaining.html