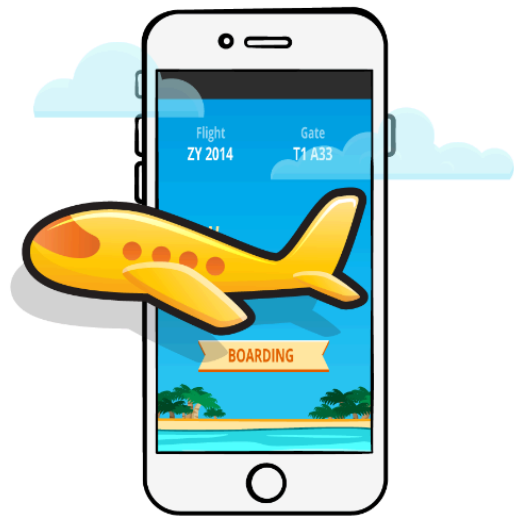


INTERMEDIATE

ios ANIMATIONS



HANDS-ON CHALLENGES

Intermediate iOS Animations

Catie & Jessy Catterwaul

Copyright ©2017 Razeware LLC.

Notice of Rights

All rights reserved. No part of this book or corresponding materials (such as text, images, or source code) may be reproduced or distributed by any means without prior written permission of the copyright owner.

Notice of Liability

This challenge and all corresponding materials (such as source code) are provided on an "as is" basis, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in action of contract, tort or otherwise, arising from, out of or in connection with the software or the use of other dealing in the software.

Trademarks

All trademarks and registered trademarks appearing in this book are the property of their own respective owners.

Table of Contents: Overview

Challenge 11: Layer Keyframes	5
-------------------------------------	---

Table of Contents: Extended

Challenge 11: Layer Keyframes	5
Keyframe Experiments	5

Challenge 11: Layer Keyframes

By Catie & Jessy Catterwaul

Keyframe Experiments

In this challenge you will create a keyframe animation for the login button. It's a little experiment, because you will learn how to tell the animation to not animate anything for a short bit in the middle. If that sounds intriguing, read onward!

Below, you will create a keyframe animation, which will fade the login button out when pressed, keep it barely visible for a while, and then fade it back in. Thanks to keyframe animations that is real easy to do.

Open **ViewController.swift** and scroll to `login()`. Append at the bottom of that method:

```
let fade = CAKeyframeAnimation(keyPath: keyPath.opacity)
fade.duration = 5
```

You define a keyframe animation on a layer's opacity and a duration of 5 seconds. Nothing too crazy in here.

Now define the key times throughout the animation that you want to define key values for:

```
fade.keyTimes = [0.0, 0.2, 0.8, 1.0]
```

So you will have the opacity animate from 0% to 10% through the animation, then to 90% and finally to 100%. Now let's add the values you want at those key times:

```
fade.values = [1.0, 0.33, 0.33, 1.0]
```

In the first 10% of the duration you will fade the layer from fully visible to 0.33 opacity, then you will keep the opacity at 0.33 until 90% through the animation. Finally, you will fade the layer back to a 1.0 opacity. See – I told you it was going to be easy!

Now you can use the same keyframe animation for both the button and the info label:

```
loginButton.layer.add(fade, forKey: nil)
info.layer.add(fade, forKey: nil)
```

Run the project and tap the button. You will see both controls fade out until they are barely visible, stay like that for a while, and then reappear in their full glory!