

iOS Animation with Swift

Hands-On Challenges

iOS Animation with Swift Hands-On Challenges

Copyright © 2014 Razeware LLC.

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.

This book 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 or other dealings in the software.

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



Challenge B: Tint animation

So far you animated x and y coordinates and that was relatively easy. For this challenge you will experiment with animating an `NSColor` property – the background color of the login button in your form.

Part 1: Animating a color

UIKit knows how to animate each of the animatable properties on a `UIView`. When such a property cannot be expressed via a simple value, UIKit still knows how to calculate all intermediate values in order to show the value change in an animated way.

So to try this out – change the login button background color from inside an animation block. Open **ViewController.swift**, scroll to the `login()` method, and inside the animations block at the bottom add:

```
self.loginButton.backgroundColor = UIColor(red: 0.85, green: 0.83,
blue: 0.45, alpha: 1.0)
```

This will tint the color of the button while it moves down and changes its bounds. Once more, if you want to observe better the intermediate states of the animation, increase the animation duration.

