

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 E: Keyframe animations

This challenge will get you to exercise the skills to create a basic keyframe animation.

Add a new method to the view controller class to define the keyframe animation for this challenge and position the status banner out of the screen bounds:

```
func animateStatusBannerWithKeyframes() {
    statusBanner.center.x -= view.bounds.size.width
    statusBanner.center.y -= 50.0
}
```

This code will move the button to the left of the screen and a bit upwards. Next add the keyframe animation to that same method:

```
UIView.animateKeyframesWithDuration(1.0, delay: 0.0, options: nil,
    animations: {

    }, completion: nil)
```

This configures the total duration and the only thing left to do is to add some keyframes from within the animations block.

First add the initial phase, in which the banner flies in the screen sideways (**NB:** make sure the code is inside the animations block):

```
UIView.addKeyframeWithRelativeStartTime(0.0, relativeDuration:
    0.33, animations: {
    self.statusBanner.center.x += self.view.bounds.size.width
    })
```

Then add the second keyframe to the animations block to move the button down to its final position:

```
UIView.addKeyframeWithRelativeStartTime(0.33, relativeDuration:
    0.66, animations: {
    self.statusBanner.center.y += 50.0
    })
```



That's it for this challenge – to give the app a test you only need to call this new method at the bottom of `changeFlightDataAnimatedTo(data:)`:

```
animateStatusBannerWithKeyframes()
```

If you want to experiment a little bit you can add more keyframes to the animation and make the button make more zig-zags until it reaches its final position on screen.

