

# WatchKit

Hands-On Challenges

# WatchKit 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: More Actions

You've seen how to wire up an action to respond to user interaction; now it's time to give it another try and get some more practice!

There are three steps to this challenge:

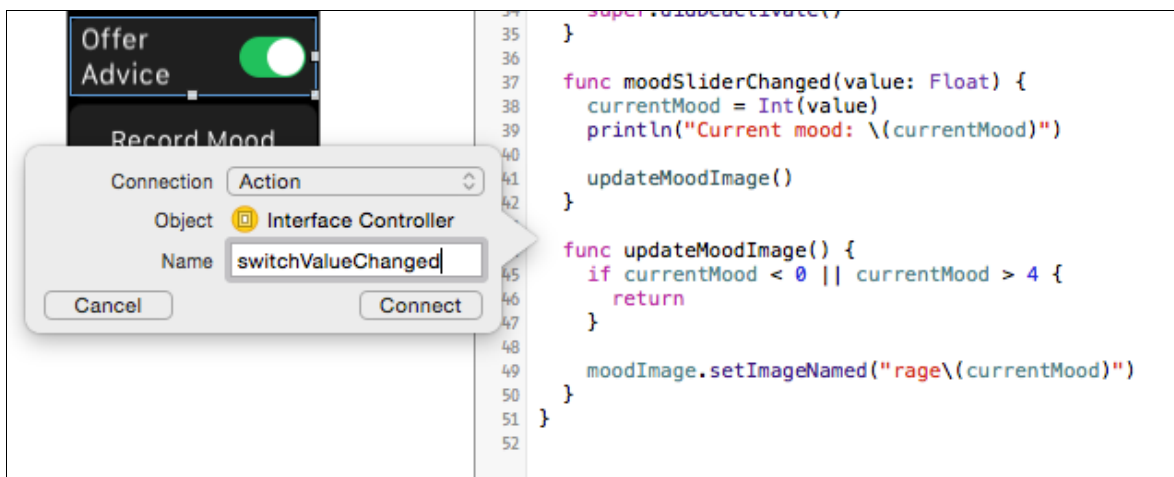
1. Wire up an action from the switch element
2. Add a property to store the value
3. Implement the action method

I encourage you to give this a try yourself! If you need some more hints or get stuck, just follow along with the solution below.

## Creating the action

Open **Interface.storyboard** and select the assistant editor. Make sure **InterfaceController.swift** is displayed in the assistant.

Control-drag from the switch to a point inside the class. Select **Action** as the connection type and name it **switchValueChanged**.



That will create a blank action method that you'll implement later.

If you see a compile error on the new method as in the demo, remove the `@IBAction` from the method declaration.



## Adding a property

Open **InterfaceController.swift** and add a new property to the class:

```
var shouldShowAdvice = true
```

This will declare a new Boolean to store the value of the switch. Note that the switch is on by default in the storyboard, so you use the same default value for the property.

## Implementing the action method

Here's where you tie it all together! Add the following implementation to `switchValueChanged:`

```
shouldShowAdvice = value  
println("Current switch value: \(shouldShowAdvice)")
```

This will update the property with the new value of the switch and then log it to the console.

Build and run, and then scroll down in the app to toggle the switch a few times. You should see the messages in the console.

Now you're all set to move on to the next video, where you'll learn about segues and how to pass the data you've been saving to the next screen!

