

# Advanced watchOS

Hands-On Challenges

# Beginning watchOS Hands-On Challenges

Copyright © 2016 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.



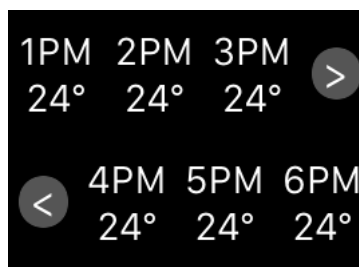
# Challenge A: Layout & Animation

Right now, your users can switch back and forth between the two “pages” of short-term forecast data. In this short challenge, you’ll tweak the layout and add a little animation to give the interface a final bit of polish.

Let’s get started!

## Side-by-side layout

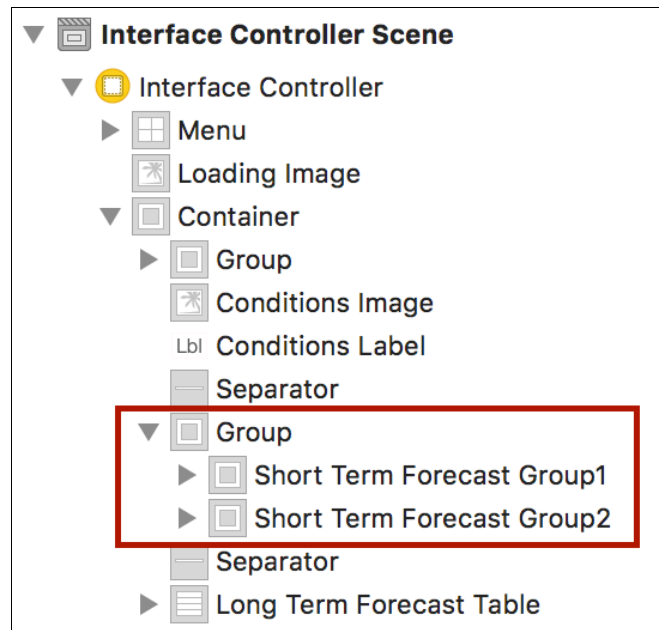
The two short-term forecast groups are currently stacked vertically.



The idea is to stack them *horizontally* instead so they’re side by side.

Start by adding a new group to the layout next to the existing short-term forecast groups. Then place the two short-term forecast groups *inside* this group.





By default, groups are already set to stack horizontally which means that's it for the layout!

Things will look a little strange in the storyboard, but rest assured the two forecast groups are now side by side as intended.



Time to head to the code — open **InterfaceController.swift** and find `awakeFromContext(_:)`. The last line of the method hides the second short-term forecast group, which you don't need to do anymore since it's off-screen.

Find that line and **delete** it:

```
shortTermForecastGroup2.setHidden(true)
```

Now it's time to look at the animation!

## Animating widths

You already saw some animation in the beginner series, and this is no different. When the user goes from page 1 to page 2, the idea here is to animate out page 1 by setting its width to 0 and fading it out. You don't need to do anything to page 2 — it will slide in from the right as page 1 shrinks down.



Still in **InterfaceController.swift**, find `showSecondPage()` and replace the implementation with the following:

```
animateWithDuration(1.0) {  
    self.shortTermForecastGroup1.setRelativeWidth(0,  
        withAdjustment: 0)  
    self.shortTermForecastGroup1.setAlpha(0)  
}
```

It's a very simple animation block: you're setting the first page's relative width to 0 to start. That's 0% relative to its container. Next, you set the alpha to 0 to fade it out. Over the one second of the animation, the group will shrink down and fade out. Easy!

Next, find `showFirstPage()`, which will need to undo the above animation to restore page 1 to its original state. Update the implementation with the following:

```
animateWithDuration(1.0) {  
    self.shortTermForecastGroup1.setRelativeWidth(1,  
        withAdjustment: 0)  
    self.shortTermForecastGroup1.setAlpha(1)  
}
```

This time you set the relative width back to 1, or 100% of the container, and the alpha to 1 to make the group fully visible.

Build and run the app, and you'll see the transition between the two groups when you hit the two buttons.

