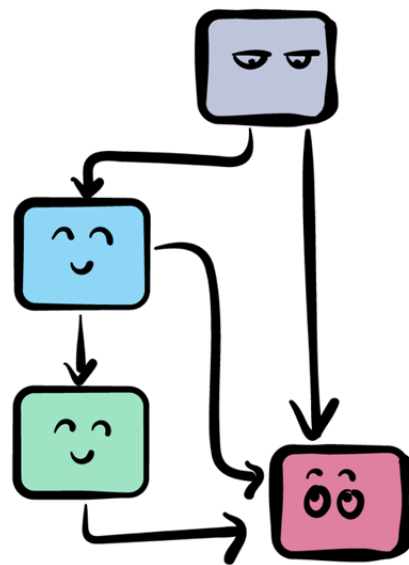


# »» iOS «« CONCURRENCY ..... WITH GCD & OPERATIONS



HANDS-ON CHALLENGES

## iOS Concurrency

Audrey Tam

Copyright ©2016 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.

## Challenge #10: Hipster Filter

The images in the TiltShift app currently look good, but not quite hipster enough - there really should be some kind of old-school camera effect applied to them after the tilt-shift filter. Luckily, the Operation-based design makes this pretty simple to implement. Your challenge is to add the new filtering operation - to achieve a result like that shown on the last slide.

Here are the steps you need to complete:

1. Open the **TiltShiftDemoFinished** version of the **TiltShift** app.
2. Use the function in **Utilities/ImageFiltering/PostProcess.swift** to create a `PostProcessImageOperation` subclass of `ImageFilterOperation` in a new file **PostProcessImageOperation.swift** in the **Operations** group.
3. Modify **Models/TiltShiftImageProvider.swift** to create a `PostProcessImageOperation` object and insert it into the dependency chain, between `tiltShift` and `filterOutput`.