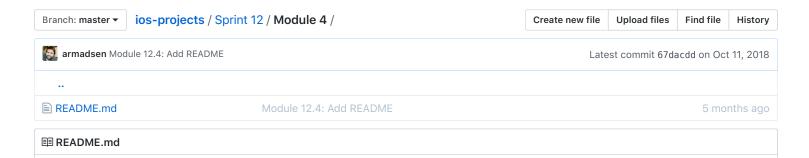
# LambdaSchool / ios-projects



# Contacts MRC

# Introduction

The goal of this project is to solidify your understanding of the reference counting memory system by writing a project using Objective-C and manual reference counting (MRC).

#### Instructions

You will need to create your own Xcode project and repository. Commit regularly as you complete the requirements in this project.

You will be implementing a basic contacts manager app. It should include:

- A main view with a table view displaying contact names
- A plus button used to create a new contact
- A detail view controller used to display, enter, and edit more information for each contact, including at least: name, email address, phone number.
- Tapping a contact should show the detail view controller and allow you to edit it.

#### Part 0 - Preparation

When you create a new project, by default, ARC is enabled. You'll need to disable ARC so you can use manual reference counting. To do so, follow these steps:

- 1. Create the project
- 2. Select the project itself in the files navigator
- 3. Select the "Build Settings" tab
- 4. Select the Project itself under "PROJECT", not the app target
- 5. Search for "objective-c automatic" which will bring up the "Objective-C Automatic Reference Counting" setting.
- 6. Change the setting to NO to disable automatic reference counting.

# Part 1 - Storyboard

Build your storyboard as you normally would. MRC makes no difference in Interface Builder, and you'll build things as you always have.

#### Part 2 - Write the Code

Implement the app code. At a high level, this code will be identical to what you're used to, except that you will be responsible for memory management. As you work, remember the 5 rules of MRC:

- 1. If you get an object from a method that starts with alloc/init, new, copy, or mutableCopy, you own it.
- 2. Otherwise, call retain to take ownership of an object.
- 3. If you own an object you must release it (or autorelease it) when you're done.
- 4. If you don't own an object you must not release it.
- 5. If you need an object to stick around longer than the current method, you must own it.

### Part 3 - Testing and Analysis

Run the static analyzer using Product->Analyze in the menu (or command-shift-B). If the analyzer finds any problems, fix them. When you're done, the static analyzer **should not report any problems**.

Run the app and test to make sure it functions correctly.

# Go Further

If you finish early or want to push yourself, here are a few additional features you can implement:

- Implement persistence using Core Data.
- · Add support for contacts having a photo. Does this effect memory usage as the app runs?