

INTERMEDIATE REALM

on ios



HANDS-ON CHALLENGES

Intermediate Realm on iOS

Marin Todorov

Copyright ©2017 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.

Table of Contents: Overview

Challenges: Encrypted Realms.....	5
-----------------------------------	---

Table of Contents: Extended

Challenges: Encrypted Realms	5
Challenge A: Add results to the app's encrypted Realm file.....	5

4 Challenges: Encrypted Realms

By Marin Todorov

Challenge A: Add results to the app's encrypted Realm file

In this challenge you will add a bit of code to your `MarksViewController` class to allow the user add their marks securely to the app's encrypted Realm file.

Open **MarksViewController.swift** and find the `addMark:` method, which is already connected to the **+** bar item in the view controller's storyboard.

First you'll need to add an alert view controller featuring two text fields; the code will be pretty similar to the one presenting the PIN dialogue. Append in `addMark:`

```
let alert = UIAlertController(title: "Add results securely",
    message: "Add subject name and your result",
    preferredStyle: .alert)
alert.addTextField(configurationHandler: nil)
alert.addTextField(configurationHandler: nil)
```

The two text fields will allow the user to enter the name of the exam and their result.

Note: In real life projects you'd like to show a list of the existing exams so that the user doesn't have to enter information they can pick from a list, but we're trying to keep UI code to a minimum for these challenges.

Next add a button to the alert and present the controller on screen:

```
alert.addAction(UIAlertAction(title: "Save", style: .default, handler:
{[weak self] action in
    //add code here
}))
```

```
present(alert, animated: true, completion: nil)
```

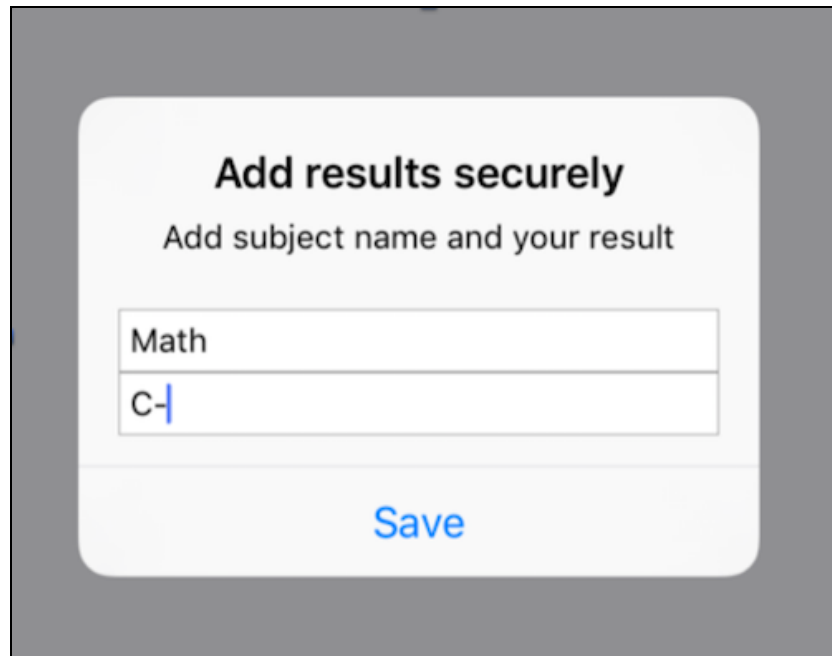
Inside the handler closure add the logic to check the user input:

```
if let subject = alert.textFields?.first?.text,
   !subject.isEmpty,
   let mark = alert.textFields?.last?.text,
   !mark.isEmpty,
   let realm = self?.safeRealm {
    try! realm.write {
        realm.add(SubjectMark(name: subject, mark: mark))
    }
    self?.tableView.reloadData()
}
```

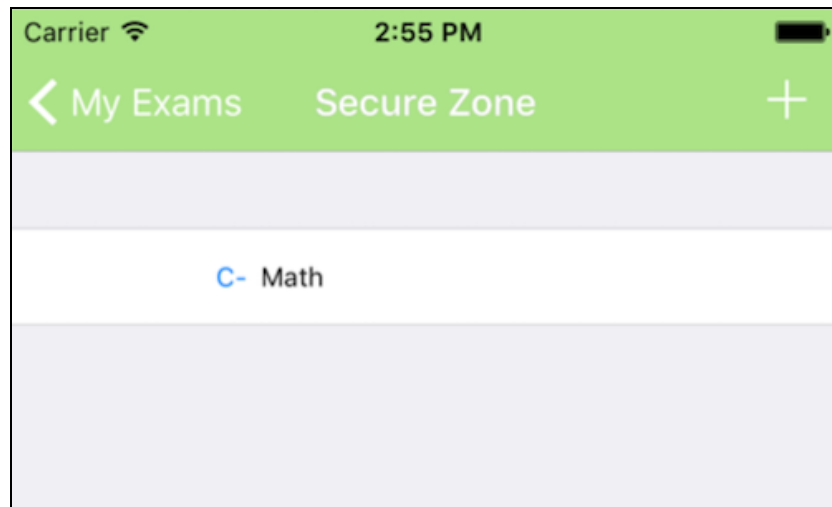
You check that the user entered some text in each of the fields. Then you store the new object in your Realm file and trigger a refresh of your table view.

You don't need to handle notifications or refresh anything else – the marks, which the table view loads its items from is always up-to-date because is a `Results` object.

When you launch the app now and enter the Secure Zone screen you can add an exam result:



And then see it immediately pop up in the table view:



Good job working through this challenge! However that C- on your math exam might require some more work :]