Create, Read, Update, Delete

I used the same approach for the Read and Update methods as we used for the ClassRoster app, namely NSKeyedArchiver and NSKeyedUnarchiver. The only new challenge was how to save the data. In the homework app I used a protocol and delegate approach where I saved the indexPath in a local variable in the calling ViewController whenever a change was made. Then in the ViewWillAppear function I called my function to save the data and used the indexPath to call reloadRowsAtIndexPaths:withRowAnimation:. This worked fine for a ViewController stack of only one after the root VC.

I created two protocols this time; one with a single function to provide the callback to the root VC to save the data. Another with a single variable to save and pass on the delegate ViewController to all levels of the VC stack. It gets used in the Book list VC for saving after deleting a book and of course in the Book Detail VC for saving after editing the title or author text field.

I had already implemented a Delete capability in the ClassRoster app, so nothing new here. This time though I kept it simple and just accepted the swipe gesture to bring up the delete option. Once again, without thinking it through, I called deleteRowsAtIndexPaths:withRowAnimation: before deleting the row in the backing array and got an exception

since the row count was now incorrect.

Implementing Create was a new iOS experience. And I have to admit to taking the easy way out. In part because I was trying to follow that principle about not adding extra functionality above the stated requirements. So I chose not to implement a Cancel. If you don't want the new book then you must delete it in the Book list TableView.

I decided to reuse the Book Detail VC for the create new operation and used a second segue from the Book list VC / NavigationItem / Add BarButtonSystemItem. Then I just created a new, empty Book instance, appended it to the books array for that shelf and then the rest was the same as for editing an existing book. This works well, but would probably require a different method if a Cancel was implemented.