Presence - Part 4

Due Date

This assignment is due by **11:59 PM on Tuesday, May 19.**

Assignment

This is the last week of the multi-part Presence project. At this point, we're fetching live data using web services, displaying it using table views, using threads to avoid hanging the user interface and allowing the user to update their own status using a modal view controller.

This week, we'll add some finishing touches. We'll add some new modes to the application, allowing the user to toggle between them by combining navigation controllers with a tab bar controller. We'll integrate with the system Address Book to better connect with our user's data. And we'll add a search feature for querying Twitter requests.

Here are the requirements for Part 4:

1. Instantiate a **tab bar controller** in your application delegate. **Add its view to the window**, rather than the navigation controller's view.
2. Tab bar controllers manage **an array of view controllers**. One of these should be our existing navigation controller, with the PersonListViewController at the bottom of the stack.
3. Create a **new view controller that displays users from the public Twitter timeline**. For this view controller, rather than reading usernames from the TwitterUsers property list, request them using the new **+[TwitterHelper fetchPublicTimeline]** method. As usual, don't block the main thread!
4. Create a **third view controller that allows searching all Twitter status updates**. Use a table view with a UISearchBar as its table header view. See the example from Lecture 14 if you're not sure how to do this. There's another new method to take care of making a search request, **+[TwitterHelper fetchSearchResultsForQuery:]**. All you need to do is display a list of matching status updates in a table view.
5. Finally, we'd like to **integrate with the system Address Book**. For each person in a person list (both for the TwitterUsers list and public timeline), customize the cell accessory type to **display a blue disclosure button**. Selecting a row will still display detailed status updates for a person, but **tapping the disclosure button should create and push an appropriate Address Book view controller**.

There is an **accompanying archive titled Presence4Files.zip** which includes an updated TwitterHelper class. You can use the original TwitterUsers property list and JSON parsing code from previous assignments. It also includes some images for your tab bar items, as well as an **application icon for Presence** which you can use if you wish.

Testing

In most assignments testing of the resulting application is the primary objective. In this case, testing/grading will be done both on the behavior of the application, and also on the code.

We will be looking at the following:

1. Your project should **build without errors or warnings** and **run without crashing**.
2. Each view controller should be the **File's Owner of its own Interface Builder document**.
3. Remember the rules of retain, release and autorelease. **Don't leak memory or over-release**.
4. Readability is important. Make sure to **decompose, comment and name thoughtfully**.
5. Your program should behave as described above. It must present (at least) three modes to the user, including one for viewing public status updates and another for searching status updates. For lists of people, display a blue disclosure button which pushes an Address Book view controller onto the navigation stack.

Hints and Reminders

A Navigation Controller is Just Another View Controller
You can use a navigation controller in most places where a view controller is expected. So, your tab bar controller's array of view controllers will include a couple of navigation controllers, each with their own navigation stack.

Avoiding Duplicated Code
The list of people from the TwitterUsers property list and the list of people from the public timeline are remarkably similar. Really, the only difference is how they fetch their list of people. You should figure out a way to share the rest of the code, possibly by creating a common superclass. We will frown heavily upon large swaths of copied & pasted code.

Displaying the Blue Disclosure Button
There is a table view delegate method that will allow you to customize the accessory view displayed for your table view cells. There is a special delegate method as well which is called when the disclosure button is pressed. See the UITableView documentation or UITableView.h

Adjusting to Keyboard Appearance
Keep the example from lecture in mind where we modified the insets of a table view for showing and hiding the keyboard. You should do this in your search view controller as well.

Don't Block the Main Thread
As usual, if you're using code which may potentially take awhile, keep it off the main thread as we did in Presence 3. Now that you're comfortable with NSOperation and NSOperationQueue, this should be easy.

Extra Credit

Here are some suggestions for extra credit:

- Add another mode (or two, or more...) to the application. You might use the included method **+[TwitterHelper fetchFriendsTimelineForUsername:withPassword:]** for one of them.
- Our requirements for the search view controller are pretty minimal- just display a list of results. Customize the appearance of the list to make it look a little nicer, and make something interesting happen when a row is selected. You may want to wrap your search view controller in its own navigation controller.
- Create a preferences bundle for your application. We haven't covered this at all in lecture, so you'll need to dig through the documentation to figure out how this works. Your preferences for the application should allow the user to set the username and password used for status updates and fetching the friends.

If you undertake any extra credit, please let us know in your submission notes or otherwise. If you don't, we might not know to look for it. And be sure that the core functionality of your application is solid before working on any of this!