iOS Assignment #2 – Advanced topics

The objective of this assignment is to get familiar with some of the more advanced features of iOS. Apps often utilize the Core Location framework and MapKit for geolocation, and almost always use asynchronous requests to third-party APIs.

In this assignment, you will build a data visualizer for Twitter, based on the location of the most recent tweets that have geolocation information. In particular, you will use the Twitter framework to get the most recent tweets within a 1-mile radius of the user's current location and display it on a map.[1] Be sure to make network requests asynchronously; synchronous requests will lock up the user interface until the request is completed![2]

But let's make this a little more interesting: the app will continuously poll for new tweets, and display them on the map when they are received.[3] Once 100 pins have been placed on the map, the 10 oldest tweets should be removed. If the app gets data for more than 100 tweets, it should display the most recent 100. Displaying too many pins on the map can slow down the event handling on the UI.

If the user taps on one of the pins, a small view should display some information about the tweet, and a button to take them to a separate view that displays more information about the tweet (user's twitter handle, the tweet's text, timestamp, user's avatar, etc.). On this separate view, the user should have the option of favoriting or retweeting that particular tweet (this can also be accomplished using the Twitter Framework).

Grading Criteria

The assignment will be graded on the following criteria:

1. User Experience
    a. Flow of the user experience
    b. Negligible latency in user interaction
    c. Error Handling
2. Efficient use of network calls and available information
    a. Asynchronous network calls
    b. Careful network calls that request only necessary data
    c. Resourceful reuse of information
3. Code quality and style

[1] Because the iOS Simulator doesn't have GPS, you'll have to spoof the location by adding a location manually. Just use Columbia's campus. When testing on a device, actual locations can be recorded. Either way, be sure to show the code that would be run if location tracking were available.

[2] This is a really important note: even if your app makes some network requests asynchronously, it **MUST** update the user interface on the main thread, or the app will crash! There are resources in GCD to accomplish this.

[3] By "continuously", I don't mean a literal open socket: maybe poll every five or ten seconds. It's up to you, as long as it's reasonable.

Resources[4]

Here are some resources that you should check out before you get started:

- Twitter Framework – This is what your app will use to handle all of its Twitter requests.
- Twitter API – This is the resource used to actually form Twitter requests, which will be handled by the Twitter Framework.
- Core Location – This is the framework that records a user's location and tracks their movement.
- MapKit – This is the framework that allows information to be plotted and manipulated on maps.
- View Controller Programming in iOS – This is a guide about building apps with multiple view controllers, and the ways to use multiple MVCs to create an intuitive and simple flow in the user experience.[5]
- iOS Human Interface Guidelines – This is hailed by iOS UI designers everywhere as the most important document in mobile user interface design. Skim through it to learn about developing beautiful applications for iOS!
- Grand Central Dispatch – This is an easy way to implement multithreaded applications, and can be used to make asynchronous calls. If you want, you can look at NSOperationQueues, but GCD will work fine for this assignment.
- JSON – JSON is a data format that's very easy to parse, and it's the format in which Twitter will return its data. If you've never used it before, read a little about it here.
- NSJSONSerialization – For those of you who didn't parse JSON in your first assignment, you'll have to in this one. This is a class built into iOS to help with that task.
- JSBeautifier – This is an online tool to format JSON so that it's more human-readable.
- Apple Sample Code – There's enough sample code here to fill a library. There are sample projects for each of the resources above if you need some help or inspiration. Just click "Sample Code" on the left hand side to access it.

As always, let me know if you have any questions. My email address is mds2161@columbia.edu, and my office hours are Tuesdays from 2:00-3:00 PM in the CS help room on the first floor of Mudd. Additionally, I'll be holding one or two more lab sessions before this assignment is due. Information about those will be sent to you when they're planned.

Best of luck!
Mason

[4] In some of these cases, there are corresponding delegates (much like the UITableViewDelegate that you dealt with in the first assignment). These are going to be very useful to you, so make sure to take a look!

[5] Hint: there are some differences between iOS 5 and iOS 6 with respect to moving between view controllers. I'm not going to tell you what they are though, you'll have to find those differences for yourselves in the documentation.