Qudini TaskApp

Benoit Pasquier

At the begining, I used to implement CocoaPods in my project because it's really simple to install libraries and update dependencies with that. However, since few dates, I've got several issues around Ruby and CocoaPods on my Mac, I couldn't implement it on TaskApp during this time.

Even if it's a small project, I always keep my files into directories with relevant names to understand quickly the structure of the project and where people can find details (controllers, design, libraries, categories or images).

Of course, I kept all the device orientations, it's more user-friendly on iPad. I only needed to handle the storyboard's constraints to keep a nice view after any rotations.

I created an UIViewController dedicated to my table, UITableViewController can be difficult to customize. I coded a manager for my web requests based on a singleton instanciation to let the manager available in other files.

I added a protocol as a delegate to handle the given events in the current controller. With this protocol, I can show an UIAlertView if an error occured or reload my tableView if all worked well.

I created an asynchronous web request with a GET HTTP methods, I didn't want to lock the UI during my requests. On the API part, it should be better to manage request with OAuth or POST methos to minimize the potential leaks.

When I got a good answer from the API, I converted my result into a NSDictionary. I extracted a NSArray which I sorted with a block function based on the customer's expectedTime.

After reloading my tableView, I instanciated a custom UITableViewCell with the current data. However, to load the gravatar icon, I wanted to use an external library like SDWebImage, but without CocoaPods it takes a some time to install it. That's why I created two block functions to manage the current imageView. First one, it's dispatch asynchronously the web request (avoiding UI lock), second one update the view in the mainQueue (UI queue).

At the end, I also integrated the Reachability library to handle the first Internet errors and avoid useless crash for the current user. I added a MD5 category for NSString, to get the right url for gravatar icon. This category doesn't manage ARC, I should edit the compiler flag in Build Phases.

I also added an navigation controller only to get a navigationBar which gives a better design to this small app. It's also to prevent next features like selected a cell to get more details about a client on a dedicated UIViewController.