Independent Study Final Report
Building an iOS app with ReactNative

Advisor: Prof. Barbara Lerner

Nachwa El khamlichi Fall 2016

Outline:

- Intro:
    - Project description & attainment.
- Motivation behind building an iOS app with ReactNative.
- ReactNative main concepts.
- Future Work.

Intro

Through this independent study, I was able to achieve two main things: work on a product I am passionate about building, as well as explore a new technology to build it.

A little background about the app I wanted to build: in short, "Candid" is a selfless Snapchat. The idea of the app came from the belief that many moments in our lives are not captured by people around us, there is a lack of incentive to take random pictures of people we know when we think that the moment is worth capturing. Whether it is capturing a friend laying on the floor in the library on finals week, or a classmate doing their thesis defense, Candid is meant to provide the platform that would incentivized us and friends surrounding us to capture our most candid moments.

After a semester of learning a new technology, and applying it to the project, the app was starting to come together piece by piece. Eventually, I ended up with a working beta that has most of the UI components and functionality desired. The backend service for this project was built by my good friend from high school Hamza Bourrahim, who I thank for all his efforts.

The timeline and progress of this project can be found here.

Motivation behind building an iOS app with ReactNative

There were two main reasons why I decided to build this app with ReactNative. The first one is that I have experience with web development. Since I enjoyed many of the practices and the building blocks of web apps, I thought that it would be very interesting and fun to be able to use a framework based on web development protocols in the making of an iOS app.

My second motivation was that ReactNative is “Learn once, write anywhere”. Which means that I only have to learn how to work with the ReactNative framework once to be able to build mobile apps on many platforms, including iOS, Android and many others. This particular characteristic of ReactNative is very appealing. Because unlike other frameworks like PhoneGap that allow you to build an app once, where the framework does the job of converting the code for each platform. This ends up making apps that are very buggy and that don’t have native components to each platform, which is the essential thing that creates a high quality interface that is both fast and highly responsive to the user. So unlike those frameworks, ReactNative provides a Javascript library to interact with native components to each platform. I had access to all the native iOS components (camera, ListView, GridView, ImageView…) and was able to interact with them through Javascript.

The other reason behind choosing ReactNative is the way I was able to structure my project, which I will explain in the next section, which is about React as a framework.

Description of React as a framework

React is a front end library developed by Facebook. It is a declarative, efficient, and flexible Javascript library for building user interfaces.

It is used for handling the view layer for web and mobile apps. It allows us to create reusable UI components. It is currently one of the most popular JS libraries and it has a large community and documentation behind it as well as a strong foundation.

The main important feature of using the React framework is the ability to bundle Javascript and the markup into JSX, which makes components easily understandable without having to trace complex flow from JS to HTML.

The ability to tie functionality and logic to markup in a self contained component makes the building of complex UIs an easier task. Also, what makes React’s building structure so convenient for building user interfaces is that data is either received from a component’s parent component, or it’s contained in the component itself.

These are the main concepts we have learnt about React/ReactNative which were implemented in our project:

JSX:

- Allows us to combine markup and JS logic in a single component.

Virtual DOM:

- React creates a JS representation of the actual DOM, then applies a diffing algorithm to find which pieces of the DOM need to be changed. React applies the changes without rebuilding the whole DOM. This makes the UI rendering very fast.

States:

- Allows us to manipulate the UI through a couple of methods:

    GetInitialState:

    - Allows us to set the initial state of each component.

    SetState:

    - Helper method to update the state of a component

Props:

- Using props, we can send data to child components, it is a powerful way to render dynamic data.

Events:

- OnClick, OnSubmit, OnChange: these are event handlers that allow us to detect a user input(interaction through the UI) and use the event to modify the UI accordingly.

A more thorough explanation of each of these concepts can be found in the workshop script that was produced for HackHolyoke.

Description of the App structure:

Candid is an app comprised of 3 main views. The camera view, the friends list view and the Candids Grid view as well as the Login view.

Our final MVP meets these main functionalities:
A user is able to :

- Login:
  - We use the Facebook API to authorize a user and we then receive a list of all of the user’s friends that have authorized app as well, which we display in the friends list view.
- take a picture of a friend through the Camera View:
  - The picture is only saved temporarily on the app in an object with other data (name of the sender, fb id of the sender & timestamp).
- Choose one or more friends to send the picture to through the Friends List view.
- Find and save the candid sent to the user:
  - The user can scroll through all the candids they have received from their friends in a Gridview, where we show a thumbnail of the actual picture.
  - The user can press on an individual picture and see its detail view, which contains the original sized picture, a small photo of the sender and a save button.

Future Work

Through this semester, we were able to build a good beta version of our app. We have built the main components and the main functionalities that meet the requirements of a working app that we were hoping to build!

But there are many things that we can improve on to have a production level app, which are planned for our future work. The main ones are:

- **Better caching:** The image detailView for example, has a very loading time when a user tries to swipe to the next image. The images are not loaded consecutively and need to be cached in a better way when they are first received by the user.
- **Ability to send a candid without a network:** In our current implementation, a user cannot send an image if they have no network. Optimally, we want to have a store and forward system that checks if the user

doesn’t have a network, then caches the images taken by the user with the corresponding receivers in a queue, and once the user has an internet connection, the queue is emptied and all pictures are sent in order.