Hassan Ndow

12/9/16

ISTE-454.01

Final Project Documentation

**Application Title:** NBAPrime

**Application Definition Statement:** NBAPrime is an app that lets you view a list of the 30 current and active NBA teams, view each individual team’s roster, view each player’s basic statistical profile, view the end result of a selected team’s last completed game with animated feedback, and view a box score displaying various stats of a selected team’s last completed game.

**Implemented Features:**

* Draggable player images with audio
* Teams with logos display
* Team Rosters display
* Roster players’ statistical bio
* Selected team game results with animation feedback
* Box Score display

**Next Version Features:**

* Matchup Bets
* Daily leaders in respective stats
* Highlight videos
* Team news/trades/etc
* Favorite teams for saved state selection

**Self Evaluation:**

I would grade my NBAPrime app project as an A considering the following A-level work I believe I’ve fulfilled:

* Use of dynamic data through JSON urls, plists and loaded images
* Use of Model-View-Controller pattern for separation of concerns
* Use of delegation/protocols
  + UIPickerViewDelegate
  + UIPickerViewDataSource
* Loosely coupled/reusable methods for loading/parsing JSON
* Decent navigation interaction/patterns:
  + Good use of complementary contrasting colors (avoids usage of green/red together for the color blind as well)
  + Use of animations based on dynamically loaded data

**Layout:**

* + UITabBarController
    - UIViewController
    - UINavigationController
      * UICollectionViewController
        + UITableViewController
    - UIViewController
    - UIViewController
* Easy to use app in any environment
* Data displayed within 1-3 taps
* use of enums for error checking json parsing
* “fails gracefully”
  + alert dialog informs user the number of seconds to wait before they can load data if they receive a 404 error from the API due to too many requests

**Classes:**

**AppDelegate**: The **AppDelegate** class loads a plist containing team names into an instantiated **NBAJsonUrl** object. It links up the app’s UITabBarController, **NBAStatTrackerVC**, UINavigationController, **NBAAnimatedVC**, **NBATeamsCollectionVC** and **NBABoxScoreVC** in order to pass the **NBAJsonUrl** object to controller classes.

**Models**

**NBAJsonUrl**: The **NBAJsonUrl** class inherits Objective-C’s NSObject class. It contains getters/setters for an array of nba team names, the common components accessed by all the classes that make use of the specified JSON url (**NBATeamsCollectionVC**, **NBATeamsDetailTableVC**, **NBAAnimatedVC**) excluding the end points. These components include an encrypted access token stored and retrieved in the keychain with the help of **KeyChainSwiftDistrib** methods, the domain name(“https://erikberg.com”), the type of sport (“nba”), and the format (.json).

**NBAPlayer**: The **NBAPlayer** class inherits Objective-C’s NSObject class. It represents an NBA player with getters/setters for a player’s name, position, age, height, and weight. It is used primarily to store a player’s information retrieved from a JSON object.

**NBARequestLimit**: The **NBARequestLimit** class inherits Objective-C’s NSObject class. It returns the calculated result of the time difference between the future wait time of when a request can be made to the xmlstats API and the user’s current time in the form of Unix seconds remaining. **NBATeamsDetailTableVC**, **NBAAnimatedVC**, and **NBABoxScoreVC** all make use of this class.

**KeyChainSwiftDistrib**: A third party class that provides a collection of helper functions for setting/getting text and data to the keychain. **NBAJsonUrl** makes use of this class to securely store its access token.

**Views**

**NBAAnimatedView**: The **NBAAnimatedView** class is a custom UIView class. It draws animations for “hidden” images and updates the images’ frames and results label text accordingly based on the text value of a global class String variable passed to it from **NBAAnimatedVC**.

**Controllers**

**NBAStatTrackerVC**: The **NBAStatTrackerVC** class inherits the UIViewController class. It stores the **NBAJsonURl** object passed by reference from **AppDelegate**, and displays basic info about the developer of this project and the JSON’s host. Clicking the host label takes you to the domain of where the json is hosted for more information about the API. It also contains images that are draggable by user touch and respond with audio as well. For all intents and purposes, this is the home screen.

**NBATeamsCollectionVC**: The **NBATeamsCollectionVC** class inherits the UICollectionViewController class. It stores the **NBAJsonURl** object passed by reference from **AppDelegate**, displays a collection of cell items containing each NBA team’s logo. Selecting an NBA team’s logo will push the navigation to **NBATeamsDetailTableVC** that will display the respective team’s roster.

**NBATeamsDetailTableVC**: The **NBATeamsDetailTableVC** class inherits the UITableViewController class. It builds the JSON url directing it towards its appropriate endpoint with specified parameters, loads and parses the JSON url, and displays a table list of a selected team’s roster with each row represented by a player’s name accessed from an array of **NBAPlayer** objects. Selecting a row will bring up an alert box providing further statistical details about the selected player.

**NBAAnimatedVC**: The **NBAAnimatedVC** class inherits the UIViewController class and adopts the UIPickerViewDataSource and UIPickerViewDelegate protocols. It stores the **NBAJsonURl** object passed by reference from **AppDelegate**, builds the JSON url directing it towards its appropriate endpoint with specified parameters, loads and parses the JSON url that returns a String value for a team’s last played game result (win/loss) which is stored in a global class variable to be passed to **NBAAnimatedView**, and populates a pickerview with the team names.

**NBABoxScoreVC**: The **NBABoxScoreVC** class inherits the UIViewController class and adopts the UIPickerViewDataSource and UIPickerViewDelegate protocols. It stores the **NBAJsonURl** object passed by reference from **AppDelegate**, builds the JSON url directing it towards its appropriate endpoint with specified parameters based on the selected item in the picker view. It loads and parses the JSON url which returns a String value for a team’s last played game unique event id (“datePlayed-awayTeamName-homeTeamName”) and is stored in a variable passed to a second json parser that populates the box score with the team names and scores per quarter period.

**Above and beyond:**

* Access token information stored securely in keychain
* Animation on images based on value returned by a dynamically populated picker view
* Customized launch screen and app icon
* Use of NSURLSession, DispatchQueue.main.async and tableView.reloadData to properly load, update and display team rosters in a table view
* alert message that informs user on number of seconds to wait before a request can be made if a 429 HTTP error code is returned
  + fail safe cross checking throughout all controllers
* attributed tableView content handled programmatically

**References:**

**Third Party libraries/APIs**

**xmlstats API** by Erik Burg

I received direct permission to use his API for class final projects and was granted access tokens by him

* <https://erikberg.com/api>
* <https://erikberg.com/api/endpoints/roster>
* <https://erikberg.com/api/endpoints/teams>
* <https://erikberg.com/api/endpoints/team-results>
* <https://erikberg.com/api/endpoints/nba-box-score>

**KeychainSwift API** by marketplacer

* <https://github.com/marketplacer/keychain-swift/blob/master/Distrib/KeychainSwiftDistrib.swift>
* <http://cocoadocs.org/docsets/KeychainSwift/7.0.0/>

**In Code References**

* <http://stackoverflow.com/questions/31805045/how-to-parse-json-in-swift-using-nsurlsession> Author: Eric Aya
* <https://www.raywenderlich.com/120442/swift-json-tutorial> Author: Attila Hegedüs
* <http://codewithchris.com/uipickerview-example/> Author: Chris Ching
* <http://stackoverflow.com/questions/28141627/how-to-convert-nshttpurlresponse-to-string-in-swift> Author: Daij-Djan
* <http://stackoverflow.com/questions/26050655/how-to-create-a-circular-button-in-swift> Author: vacawama
* <http://stackoverflow.com/questions/33316965/dragging-images-around-the-screen-using-swift> Author: vacawama