Assignment 1: Matchismo

Objective

This assignment starts off by asking you to recreate the demonstration given in the second lecture. Not to worry, the posted slides for that lecture contain a detailed walk-through. It is important, however, that you understand what you are doing with each step of that walk-through because the rest of the assignment requires you to add a simple enhancement and your assignment for next week will extend Matchismo even further.

Be sure to check out the **Hints** section below!

Materials

- Before you start this assignment, you will need to download and install Xcode 5 using the App Store on Mac OSX (*not* the App Store on your iOS device, the App Store on your *Mac*). It is free.
- The slides for all lectures can be found in the same place you found this document.
- You will also need these four images (Stanford logo and blank card in both normal and high-resolution). A simple way to download them is to click on each one of the links below (which will bring the corresponding image up in a browser window), then drag the image out of the browser window into the Finder or onto your desktop.

  stanford.png

  stanford tree.png

  blank card rounded corner.png

  blank card hi-res.png

Required Tasks

1. Follow the detailed instructions in the lecture slides (separate document) to build and run Matchismo in the iPhone Simulator in Xcode 5. Do not proceed to the next steps unless your card flips back and forth as expected and builds without warnings or errors.
2. Be sure to include all the code for the four classes shown in the first two lectures: `Card`, `Deck`, `PlayingCard` and `PlayingCardDeck`. You must *type this code in* (e.g., do not copy and paste it from somewhere). The whole point of this Required Task is to gain experience editing code in Xcode and to get used to Objective-C syntax.
3. Add a private property of type `Deck *` to the `CardGameViewController`.
4. Use lazy instantiation to allocate and initialize this property (in the property's getter) so that it starts off with a full deck of `PlayingCard`s.
5. Matchismo so far only displays the A♣ over and over. Fix Matchismo so that each time the card is flipped face up, it displays a different random card drawn from the `Deck` property you've created above. In other words, Matchismo should flip through an entire deck of playing cards in random order, showing each card one at a time.
6. Do not break the existing functionality in Matchismo (e.g. the Flips counter should still continue to count flips).

Hints

These hints are not required tasks. They are completely optional. Following them may make the assignment a little easier or better (no guarantees though!).

1. Even though the type of the property you must add is required to be a `Deck` (not `PlayingCardDeck`) you'll obviously have to lazily instantiate it using a `PlayingCardDeck`. This is perfectly legal in object-oriented programming because a `PlayingCardDeck` inherits from `Deck` and thus it "is a" `Deck`. If you are confused by this concept in object-oriented programming, this course may be rather difficult for you.
2. It doesn't make much sense for your application to come up showing the A♣. It's probably better to make it come up showing the back of the card instead.
3. A good solution will have given some thought to what happens if every card in the deck has been shown and the user still keeps flipping. Do something simple and sensible (for example, you should not have to modify the classes in the Model in any way).
4. Economy is valuable in coding: the easiest way to ensure a bug-free line of code is not to write the line of code at all. This assignment requires very, very, very few lines of code so if you find yourself writing more than a handful of lines of code (over and above what was demonstrated in lecture), you are on the wrong track.

Things to Learn

Here is a partial list of concepts this assignment is intended to let you gain practice with or otherwise demonstrate your knowledge of.

1. Getting Xcode 5 installed.
2. Interacting with Xcode 5's UI (via the walkthrough).
3. Experiencing Objective-C (the Model classes and the Controller class).
4. Getting a first look at some UIKit classes (`UIButton` and `UILabel`).
5. Gaining experience with the iOS 7 Documentation through Xcode.
6. Understanding how to add images to a project (`Images.xcassets`).
7. Seeing an example of a Model separate from a UI (Controller and its View).
8. Understanding how to use a `@property` (a random card's `contents` and `self.deck`).
9. Declaring a property (`deck`).
10. Performing lazy instantiation (the `deck`).
11. Understanding `nil` (`drawRandomCard` returns `nil` when a `deck` is empty).
12. Not "overthinking" solutions (demonstrating a simple "out of cards" solution).

Evaluation

In all of the assignments this quarter, writing quality code that builds without warnings or errors, and then testing the resulting application and iterating until it functions properly is the goal.

Here are the most common reasons assignments are marked down:

- Project does not build.
- Project does not build without warnings.
- One or more items in the **Required Tasks** section was not satisfied.
- A fundamental concept was not understood.
- Code is sloppy and hard to read (e.g. indentation is not consistent, etc.).
- Your solution is difficult (or impossible) for someone reading the code to understand due to lack of comments, poor variable/method names, poor solution structure, poor use of whitespace, etc.
- Assignment was turned in late (you get 3 late days per quarter, so use them wisely).

Often students ask "how much commenting of my code do I need to do?" The answer is that your code must be easily and completely understandable by anyone reading it. You can assume that the reader knows the iOS 7 SDK, but should not assume that they already know the (or a) solution to the problem.

Extra Credit

This section usually contains a few ideas for some things you could do to get some more experience with the SDK at this point in the game. However, there is no extra credit available this week since we're just getting started.