Задание 2. Set

Цель:

Цель задания состоит в том, чтобы дать вам возможность создать свое первое приложение полностью с “нуля” и самостоятельно. Оно достаточно похоже на Задание 1, которое помогло обрести вам уверенность, но достаточно отличающееся, чтобы дать вам полную свободу для накопления опыта!

Так как цель этого Задания - создать приложение с “нуля”, не начинайте с кода Задания 1, начинайте с New → Project в Xcode.

Вы захотите освежить в памяти правила игры Set.

Обязательные пункты задания

1. Реализуйте игру Set в версии соло (для одного игрока).
2. Разместите на экране по крайней мере 24 карты игры Set. В Set все карты всегда лежат “лицом” вверх.
3. При старте сдайте только 12 карт. Они могут появиться где угодно на экране (то есть необязательно их выравнивать по верху или по низу экрана или как-то еще; при старте они могут быть рассеяны, если хотите), но они не должны перекрываться.
4. Вам необходимо также иметь кнопку “Deal 3 More Cards” (Сдай еще 3 карты) (согласно правилам игры Set).
5. Разрешите пользователю выбирать карты касанием для того, чтобы попытаться составить Set. На ваше усмотрение, как показывать “выбор” в вашем UI. Некоторые идеи того, как это можно сделать представлены ниже в подсказках. Также обеспечьте возможность перехода из состояние “выбрано” (selected) в состояние “не выбрано”(deselected) (но только когда 1 или 2 (не 3) карты выбраны в данный момент).
6. После того, как выбраны 3 карты, вы должны дать пользователю индикацию, совпали ли эти 3 карты или нет (согласно правилам игры Set). Вы можете сделать это с помощью цвета или как хотите, но пользователю должно быть понятно, совпали эти 3 карты или нет...
7. Когда выбрана новая карта и есть уже 3 выбранных (selected) и не совпавших Set карты, сделайте эти 3 не совпавших карты не выбранными (deselected), а новую карту выбранной (selected).
8. Согласно правилам игры Set, когда выбрана новая карта и есть уже 3 совпавших (matching) и выбранных (selected) Set карты, замените эти 3 совпавших (matching) Set карты новыми из колоды в 81 Set карту (опять, смотрите правила игры Set и что собой представляет колода Set карт). Если колода пуста, то совпавшие (matching) Set карты не могут быть заменены, но они могут быть скрыты (hidden) в вашем UI. Если вновь выбранная карта является одной из 3-х совпавших (matching) Set карт, то никакие карты не должны быть выбранными (selected) (так как вновь выбранная карта либо будет заменена, либо будет больше невидима на UI).
9. Когда кнопка “Deal 3 More Cards” (Сдай еще 3 карты) нажата, то либо a) происходит замена выбранных карт, если они совпали, либо b) добавляются 3 карты в игру.
10. Кнопка “Deal 3 More Cards” (Сдай еще 3 карты) должна быть недоступна, если a) больше нет карт в Set колоде или b) больше нет места на UI, чтобы принять еще 3 карты (заметьте, что всегда есть место для размещения еще 3-х карт, если выбранные в данный момент карты совпали (match), так как они заменяются).
11. Вместо рисования Set карт в классической форме (мы будем это делать на следующей неделе), мы будем использовать 3 символа (▲●▪) и использовать атрибуты в NSAttributedString для соответствующего их рисования (то есть цвета и затенение

(shading). И таким образом, ваши карты могут быть просто кнопками UIButtons. Смотри подсказки с предложениями о том, как показывать различные варианты Set карт.

12. Используйте метод, который берет в качестве аргумента замыкание как значимую часть вашего решения.

13. Используйте перечисление enum как значимую часть вашего решения.

14. Добавьте осмысленное расширение extension к некоторым структурам данных как значимую часть вашего решения.

15. Ваш UI должен иметь прекрасно расположенные UI элементы и хорошо выглядеть (по крайней мере в портретном режиме, желательно также и в ландшафтном режиме, хотя это не обязательно) на любом iPhone 7 или старше. Это означает, что вам следует использовать несколько простых приемов работы с Autolayout, включая Stack Views.

Подсказки (Hints)

1. Вы можете использовать тот же самый механизм расположения UI, который мы использовали в Concentration (то есть Stack Views). В начальной стадии игры некоторые кнопки начинают с того, что не показывают карту (так как в начале у нас только 12 карт, но вы должны иметь достаточно места для размещения 24 карт) , а в более поздней стадии игры будут кнопки, представляющие совпавшие карты, которые не могут быть заменены. Мы будем обращаться с этим также, как обращались в Concentration с “matched and removed” (совпавшей и удаленной) картой (то есть кнопка существует, но она не видна пользователю).

2. Заметьте, вам не требуется выравнивать карты, когда их меньше, чем максимальное количество карт, которые можно показать, так что случайное позиционирование элементов Outlet Collection не является проблемой.

3. Есть пара действительно прекрасных методов в массиве Array. Это index(of:) и contains(). Но они работают только с теми массивами Arrays, элементы которых реализуют протокол Equatable (как это делают Int и String). Если вы хотите разместить в массиве Array свои собственные структуры данных и воспользоваться index(of:) и contains(), то просто заставьте ваш ТИП данных реализовать Equatable.

4. Мы отслеживали состояния “лицом вверх” (faceUp) и “совпала” (isMatched) в игре Concentration в наших Cards. Хотя это было здорово для демонстрации того, как изменяемость (mutability) работает в Value Type, но это, возможно, не самая лучшая архитектура. Если у вас структуры данных полностью неизменяемые (immutable) (то есть нет переменных vars, одни только константы lets), то это может сделать ваш код более понятным. Например, в реализации игры Set, возможно, легче отслеживать список всех выбранных (selected) карт (или всех уже совпавших карт), чем иметь Bool переменную var в вашей структуре данных для Set Card. Возможно, вы обнаружите, что код также стал намного проще.

5. Если вы используете подход в подсказке 4, то вам почти наверняка стоит обратить внимание на подсказку 3.

6. Вы можете показать выбор (selection), используя цвет фона backgroundColor кнопки UIButton, если хотите, но UIKit также знает, как разместить границу вокруг любого UIView (включая UIButton) с помощью следующего кода (который будет рисовать границу шириной 3 points голубым цветом, например):

```
button.layer.borderWidth = 3.0
button.layer.borderColor = UIColor.blue.cgColor
```

7. Вы можете также закруглить края вашей кнопки с использованием следующего механизма:

```
button.layer.cornerRadius = 8.0
```

8. Если вы хотите закрасить (fill) символ в NSAttributedString, то используйте NSAttributedStringKey.strokeWidth с отрицательным числом.

9. Для того, чтобы Set карта выглядела “заштрихованной” (“striped”), просто используйте

NSAttributedStringKey.foregroundColor с 15% alpha (создается с помощью UIColor метода withAlphaComponent). Цвет foregroundColor со 100% alpha может быть использован для “закрашенной“ (“filled”) карты и позитивное значение strokeWidth для “пустой” (“outline”) карты.

10. Помимо указанных выше двух атрибутов NSAttributedStringKeys, вам еще может понадобиться только NSAttributedStringKey.strokeColor.

11. Вы можете использовать какие хотите цвета для вашего UI (то есть вы не обязаны использовать “стандартные” цвета игры Set).

12. Будьте внимательны со шрифтом, который вы выбрали для кнопок с вашими Set картами. У некоторых шрифтов эти три фигуры (▲●■) могут иметь различный размер. Похоже, что у systemFont размеры всех фигур одинаковые.

13. Убедитесь, что вы в точности использовали эти три Unicode символа: (▲●■). Другие, похожие символы фигур могут не закрашиваться и не обводиться так, как нам нужно для наших целей.

14. Заметьте, что заголовок title кнопки UIButton и его attributedTitle могут устанавливаться раздельно (attributedTitle имеет приоритет, если он устанавливается) так что, например, если вы хотите, чтобы кнопка UIButton совсем не имела заголовка, то нужно установить их ОБА в nil.

15. NSAttributedStringKeys типа foregroundColor не подходят для использования в Model (заметьте, что цвет - это UIColor, что означает, что это UI элемент). Не используйте NSAttributedStrings в вашей Model.

16. Было бы хорошо иметь MVC дизайн, который бы не привязывал жестко специальные имена цветов и фигур (типа diamond (ромб) или oval (овал) или green (зеленый) или striped (штрихованный)) к именам свойств в коде нашей Model. Как видите, в этом ДЗ (где мы используем (▲●■) вместо стандартных фигур и “затенение” (shading) вместо “штриховки” (striping) и т.д.) цвета, фигуры и т.д. действительно являются UI концепцией и не должны иметь ничего общего с Model.

17. На следующем ДЗ мы совсем не будем использовать строки с атрибутами, но если вы корректно сконструировали Model на этой неделе, то ваша Model не потребует изменения даже строчки кода. Подумайте, как сделать так, чтобы ваша Model просто имела правильный API для оповещения о том, что происходит в вашей игре, а не делала бы каких-то предположений о том, как игра представляется пользователю.

18. У игры Set есть список карт, которые находятся в игре, у нее есть несколько выбранных (selected) карт, она знает, совпадают (match) или нет выбранные (selected) в настоящий момент карты, у нее есть колода карт, из которой карты сдаются (deal), и, возможно, она хочет отслеживать, какие карты уже совпали (matched). В ней действительно очень много чего. API вашей Model должно представлять все эти концепции очень ясно. Единственная действительная функциональность вашей Model состоит в выборе (selecting) карт для попытки обеспечить их совпадение и сдаче 3-х новых карт по требованию (потому что то фундаментальная концепция игры Set).

19. Предотвратить добавления еще 3-х карт, когда UI - “полностью занят” - это UI вещь (НЕ Model). Проверка доступности кнопки “Deal 3 More Cards” должна выполняться полностью в UI. У Model нет такой концепции “больше карт не помещаются в UI”, так как она ничего не знает о UI. Другими словами, Обязательный пункт 9 - это работа Model, а обязательный пункт 10 - это работа Controller.

20. Внимательно проверяй “конец игры.” Когда колода Set карт исчерпалась, успешно совпавшие карты не подлежат замене новыми картами. Эти НЕ-заменяемые совпавшие карты не могут появляться на UI (иначе пользователи могут попытаться найти совпадение их с другими картами!). По этой причине API вашей Model должен выявлять, какие карты уже успешно совпали.

21. Помните, что ваша Model реально не должна метить успешно совпавшие карты как “совпавшие” (matched) до тех пор, пока не произойдет выбор следующей карты или не сработает кнопка “Deal 3 More Cards” (что хорошо, что вы не хотите убирать совпавшие карты с UI до тех пор, пользователь не получил шанса увидеть, что он или она добились успешного совпадения карт).

22. Для проверки окончания игры, возможно, вы захотите временно обхитрить вашу Model, заставив ее думать, что любые 3 карты совпадают, так что вы сможете быстро достичь конца игры (если вы, конечно, не реально очень удачливы и искусны в игре Set!).

23. При реализации Model лучше обойтись кодом менее есть 100 строк (не считая

комментариев и фигурных скобок на отдельной строке). Но в действительности вам понадобится гораздо меньшее число строк. Ваш UI (то есть ваш ViewController) , возможно, будет иметь такой же размер кода. Если вы начинаете выходить за 200 строк кода при исполнении обязательных пунктов этого задания, то, возможно, вы двигаетесь в неверном направлении.

Что нужно изучать

Это частичный список концепций, в которых это задание увеличивает ваш опыт работы или
демонстрирует ваши знания.

1. Все, что было в Задании 1, но на этот раз с нуля.
2. Замыкания (Closures)
3. Расширение extension
4. Использование структур для декларирования констант
5. Equatable
6. enum

Дополнительные пункты (Extra Credit)

Мы постарались создать Дополнительные задания так, чтобы они расширили ваши познания в том, что мы проходили на этой неделе. Попытка выполнения по крайней мере некоторых из них приветствуется в плане получения максимального эффекта от этого курса. Сколько вы заработаете на дополнительных пунктах, зависит от контекста дополнительного пункта.

1. Фактор “скорость игры" включается в счет пользователя. Вы можете определить сколько прошло времени (в долях секунды), используя структуру struct Date. Чем быстрее пользователь находит Sets, тем больше его или ее счет должен быть. Штраф для “несовпадение”, возможно, должен быть больше, чтобы отбить охоту слишком частому простому угадыванию.
2. Штрафуйте нажатие кнопки “Deal 3 More Cards”, если в видимых картах есть доступный
Set. Вам нужно написать алгоритм для определения того, содержит ли кучка Set карт по крайней мере один Set.
3. Если вы напишете алгоритм определения Sets, то сможете добавить “мошенническую” кнопку для того, чтобы испытывающий трудности пользователь смог использовать ее для нахождения Set!
4. Или даже у вас может быть режим “играй против iPhone”. Каждый раз, когда Set обнаружен, стартуйте таймер со случайной продолжительностью (вам придется для этого изучить Timer.scheduledTimer(withTimeInterval:repeats:) { }, использующий замыкание), по истечение которой iPhone выбирает Set, если пользователь еще не выбрал. Смотрите, кто набрал больше Sets! Может быть представить iPhone с помощью эмоджи где-то на экране. Как всегда, убедитесь, что вы очень внимательно продумали свою MVC архитектуру, если вы используете этот шаблон конструирования.