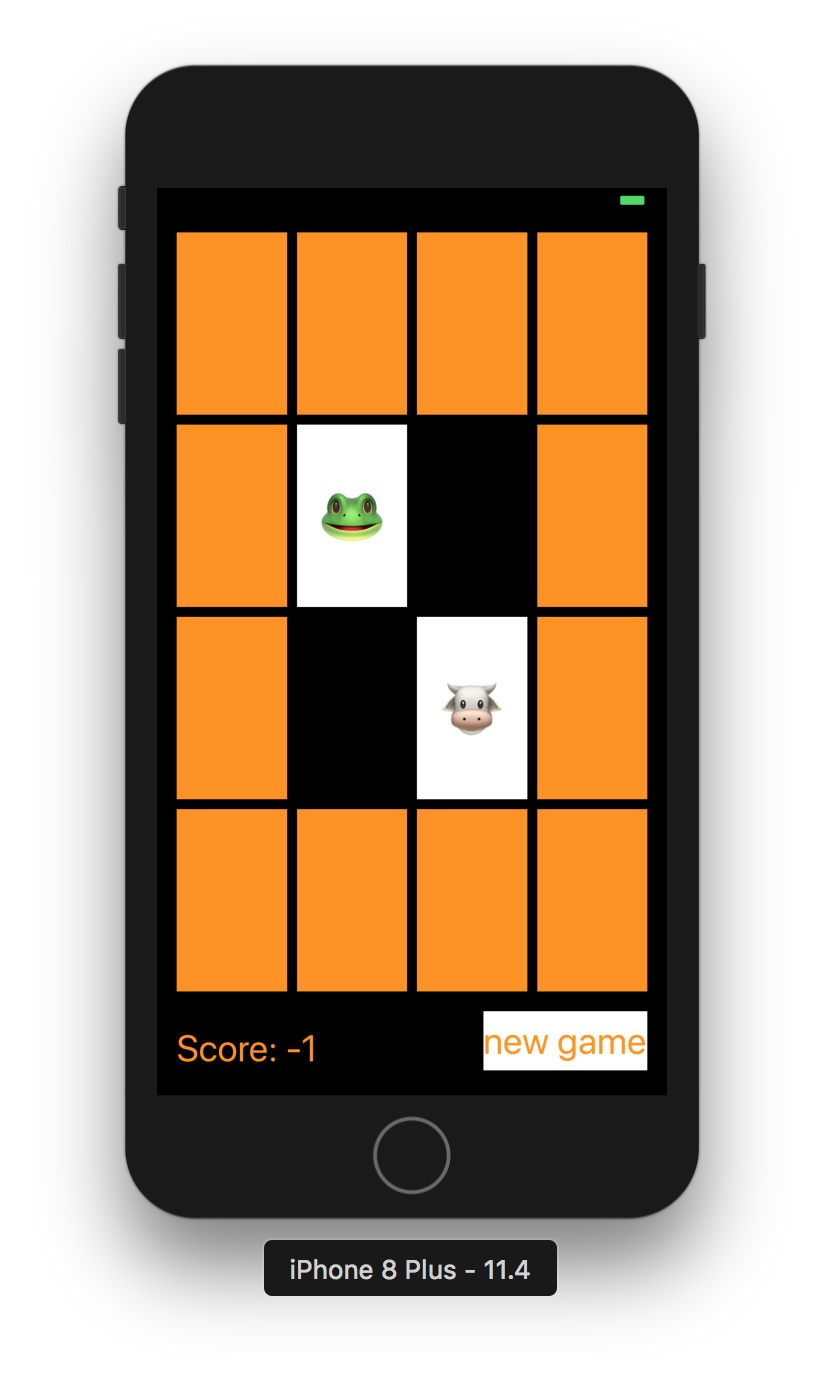
实验编号： 7 **四川师大《IOS》实验报告 2018** 年 **10** 月 **24** 日

### **计算机科学学院** 2016 级 4 班 实验名称： Game单MVC \_

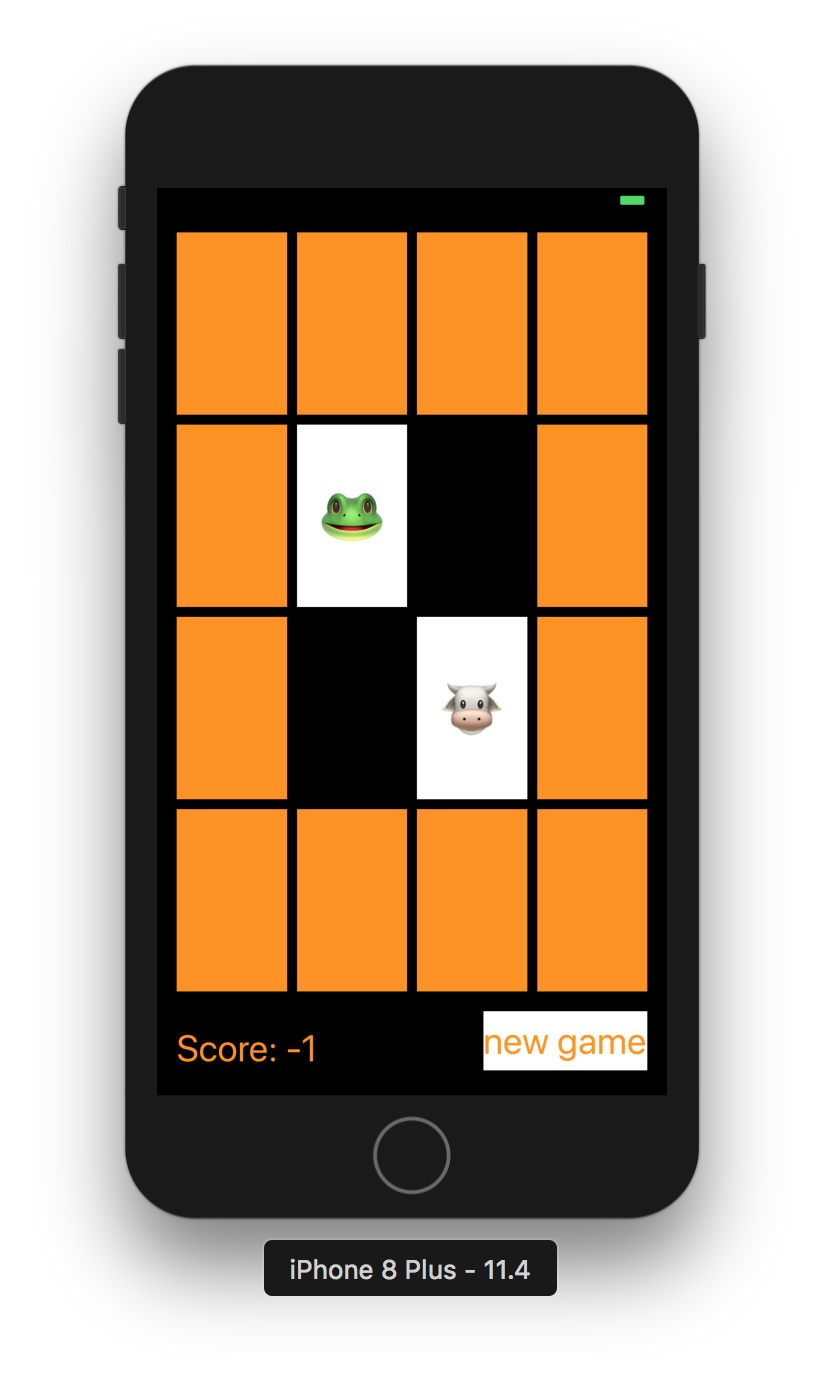
姓名：\_\_\_蒋宇童\_\_\_\_\_\_ 学号：\_\_2016110419\_\_\_\_\_\_\_\_ 指导老师：\_\_李贵洋\_\_ 实验成绩:\_\_\_\_\_

**实验 七 \_\_\_\_\_\_** Game单MVC **\_\_\_\_\_\_\_\_**

1. 实验目的及要求
2. 实现一款功能完整的game（Concentration）；
3. 掌握单MVC的主要思想；
4. 实验内容
5. 参照Stanford视频1和2完成一个game（Concentration）的制作；
6. 在(1)的基础上进一步完成Stanford Assignment 1的完整要求；
7. 采用autolayout布局解决横竖屏自适应如下所示；

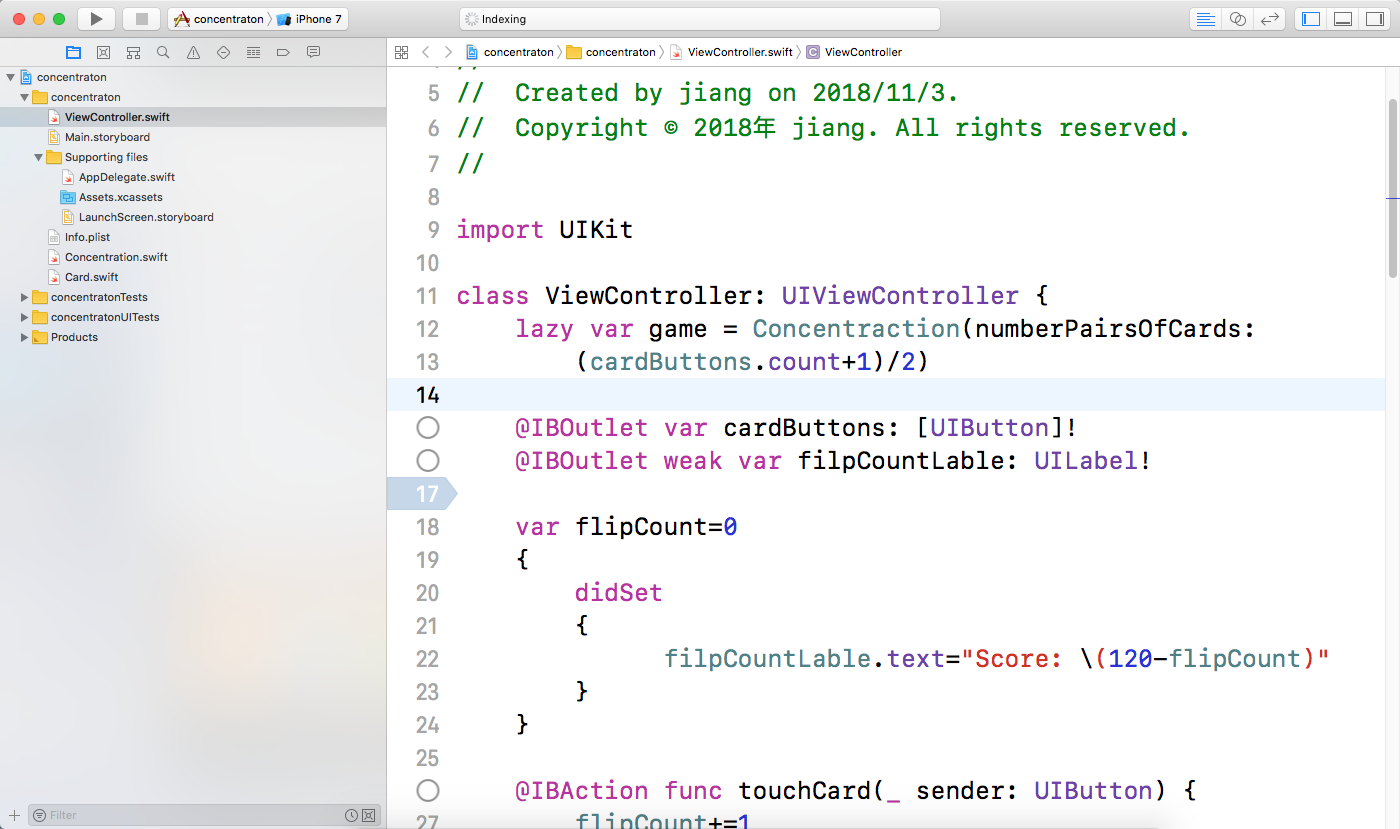


1. 实验主要流程、基本操作或核心代码、算法片段（该部分如不够填写，请另加附页）
2. 参照Stanford视频1和2完成一个game（Concentration）的制作；
3. 在(1)的基础上进一步完成Stanford Assignment 1的完整要求；
4. 采用autolayout布局解决横竖屏自适应如下所示；



* 程序代码：

左边列表文件



不同文件里的代码

import UIKit

class ViewController: UIViewController {

lazy var game = Concentraction(numberPairsOfCards:

(cardButtons.count+1)/2)

@IBOutlet var cardButtons: [UIButton]!

@IBOutlet weak var filpCountLable: UILabel!

var flipCount=0

{

didSet

{

filpCountLable.text="Score: \(120-flipCount)"

}

}

@IBAction func touchCard(\_ sender: UIButton) {

flipCount+=1

if let cardNumber=cardButtons.index(of: sender)

{

game.chooseCard(at: cardNumber)

updateViewFromModel()

}

else

{

print("there is no card")

}

}

func updateViewFromModel() {

for index in cardButtons.indices

{

let button = cardButtons[index]

let card = game.cards[index]

if card.isFaceUp

{

button.setTitle(emoji(for:card), for: UIControlState.normal)

button.backgroundColor = UIColor.white

}else

{

button.setTitle("", for: UIControlState.normal)

button.backgroundColor = card.isMatched ? colorLiteral(red: 0.9610558152, green: 0.5509537458, blue: 0.01276976243, alpha: 0): colorLiteral(red: 0.9610558152, green: 0.5509537458, blue: 0.01276976243, alpha: 1)

}

}

}

var item = [0:["🎃","👻","🧙‍♀️","🐳","😈","👹","🤖","🐹"],

1:["🍣","🍜","🍤","🍡","🍱","🎂","🍨","🍜"],

2:["⚽️","🏐","🎱","⛸","⛷","🥋","🧗🏻‍♂️","⛹🏼‍♀️"],

3:["🚇","✈️","🚖","🚈","🚁","🏍","🚔","🚀"],

4:["💗","💝","💔","💕","💘","💞","💓","❣️"],]

lazy var emojiChoices = item[0]!

var emoji = [Int:String]()

func emoji(for card:Card)->String

{

if emoji[card.identifer]==nil,emojiChoices.count>0

{

let randomInex = Int(arc4random\_uniform(UInt32(emojiChoices.count)))

emoji[card.identifer]=emojiChoices.remove(at: randomInex)

}

return emoji[card.identifer] ?? "?"

}

@IBAction func palyAgain(\_ sender: UIButton) {

game = Concentraction(numberPairsOfCards: (cardButtons.count+1)/2)

let itemnum = Int( arc4random\_uniform(UInt32(item.keys.count)))

emojiChoices = item[itemnum]!

updateViewFromModel()

flipCount=0

}

}

//

// Concentration.swift

// concentraton

//

// Created by jiang on 2018/11/20.

// Copyright © 2018年 jiang. All rights reserved.

//

import Foundation

class Concentraction

{

var cards=[Card]()

var indexOfOneAndOnlyFaceUpCard:Int?

{

get

{

var foundIndex:Int?

for index in cards.indices

{

if cards[index].isFaceUp

{

if foundIndex == nil

{

foundIndex = index

}else

{

return nil

}

}

}

return foundIndex

}

set

{

for index in cards.indices

{

cards[index].isFaceUp = (index==newValue)

}

}

}

func chooseCard(at index:Int) {

if !cards[index].isMatched

{

if let matchIndex = indexOfOneAndOnlyFaceUpCard,matchIndex != index

{

if cards[matchIndex].identifer == cards[index].identifer

{

cards[matchIndex].isMatched = true

cards[index].isMatched=true

}

cards[index].isFaceUp = true

}

else

{

for filpDownIndex in cards.indices

{

cards[filpDownIndex].isFaceUp = false

}

cards[index].isFaceUp = true

}

}

}

init(numberPairsOfCards:Int) {

for \_ in 1...numberPairsOfCards

{

let card = Card()

cards += [card,card]

}

// TODO: Shuffle the cards

cards.sort { \_,\_ in arc4random\_uniform(2) > 0 }

}

}

//

// Card.swift

// concentraton

//

// Created by jiang on 2018/11/20.

// Copyright © 2018年 jiang. All rights reserved.

//

import Foundation

struct Card {

var isFaceUp = false

var isMatched = false

var identifer:Int

static var identfierFactory = 0

static func getUniqueIdentifer()->Int

{

identfierFactory += 1

return identfierFactory

}

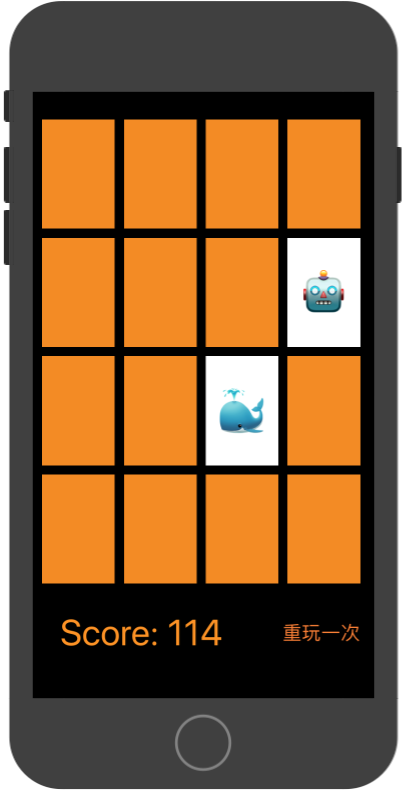
init() {

self.identifer = Card.getUniqueIdentifer()

}

}

* 运行截图：





1. 实验结果的分析与评价（该部分如不够填写，请另加附页）

**Github地址：**[**https://github.com/jiangyutong/swiftWork/tree/master/代码**](https://github.com/jiangyutong/swiftWork/tree/master/代码)

这个实验是做一个翻纸牌的小游戏，就是相同的就可以消失。我设计的分数规则是最开始的时候呢，分数是120分，每当玩家翻一次牌就分数就减一分。该游戏设置了约束不管是横屏还是竖屏元素都不会乱。

注：实验成绩等级分为（90－100分）优，（80－89分）良，(70-79分)中，（60－69分）及格，（59分）不及格。