```
//----------------------//----------------------
import UIKit
//----------------------//----------------------
class ViewController: UIViewController {
    @IBOutlet weak var tempLabel: UILabel!
    //---
    @IBOutlet weak var slot_1: UIImageView!
    @IBOutlet weak var slot_2: UIImageView!
    @IBOutlet weak var slot_3: UIImageView!
    @IBOutlet weak var slot_4: UIImageView!
    @IBOutlet weak var slot_5: UIImageView!
    //---
    var card_blur_1: UIImage!
    var card_blur_2: UIImage!
    var card_blur_3: UIImage!
    var card_blur_4: UIImage!
    var card_blur_5: UIImage!
    //---
    @IBOutlet weak var bg_1: UIView!
    @IBOutlet weak var bg_2: UIView!
    @IBOutlet weak var bg_3: UIView!
    @IBOutlet weak var bg_4: UIView!
    @IBOutlet weak var bg_5: UIView!
    //---
    @IBOutlet weak var keep_1: UILabel!
    @IBOutlet weak var keep_2: UILabel!
    @IBOutlet weak var keep_3: UILabel!
    @IBOutlet weak var keep_4: UILabel!
    @IBOutlet weak var keep_5: UILabel!
    //---
    @IBOutlet weak var dealButton: UIButton!
    @IBOutlet weak var creditsLabel: UILabel!
    @IBOutlet weak var betLabel: UILabel!
    //---
    var arrOfCardImages: [UIImage]!
    //---
    var arrOfSlotImageViews: [UIImageView]!
    //---
    var deckOfCards = [(Int, String)]()
    //---
    var arrOfBackgrounds: [UIView]!
    //---
    var arrOfKeepLabels: [UILabel]!
    //---
    var permissionToSelectCards = false
    var bet = 0
    var credits = 2000
    //---
    var chances = 2
    //---
    let pokerHands = PokerHands()
    //---
    var handToAnalyse = [(0, ""), (0, ""), (0, ""), (0, ""), (0, "")]
    //---
```

```swift
var theHand = [(Int, String)]()
//----------------------//----------------------
override func viewDidLoad() {
    //---
    super.viewDidLoad()
    //---
    createCardObjectsFromImages()
    //---
    fillUpArrays()
    //---
    prepareAnimations(duration: 0.5,
                      repeating: 5,
                      cards: arrOfCardImages)
    //---
    stylizeSlotImageViews(radius: 10,
                          borderWidth: 0.5,
                          borderColor: UIColor.black.cgColor,
                          bgColor: UIColor.yellow.cgColor)
    //---
    stylizeBackgroundViews(radius: 10,
                           borderWidth: nil,
                           borderColor: UIColor.black.cgColor,
                           bgColor: nil)
    //---
    createDeckOfCards()
    //---
}
//----------------------//----------------------
func createDeckOfCards() {
    deckOfCards = [(Int, String)]()
    for a in 0...3 {
        let suits = ["d", "h", "c", "s"]
        for b in 1...13 {
            deckOfCards.append((b, suits[a]))
        }
    }
}
//----------------------//----------------------
func stylizeSlotImageViews(radius r: CGFloat,
                           borderWidth w: CGFloat,
                           borderColor c: CGColor,
                           bgColor g: CGColor!) {
    for slotImageView in arrOfSlotImageViews {
        slotImageView.clipsToBounds = true
        slotImageView.layer.cornerRadius = r
        slotImageView.layer.borderWidth = w
        slotImageView.layer.borderColor = c
        slotImageView.layer.backgroundColor = g
    }
}
//----------------------//----------------------
func stylizeBackgroundViews(radius r: CGFloat,
                            borderWidth w: CGFloat?,
                            borderColor c: CGColor,
                              bgColor g: CGColor?) {
        for bgView in arrOfBackgrounds {
            bgView.clipsToBounds = true
            bgView.layer.cornerRadius = r
            bgView.layer.borderWidth = w ?? 0
            bgView.layer.borderColor = c
            bgView.layer.backgroundColor = g
        }
    }
    //----------------------//----------------------
    func fillUpArrays() {
        arrOfCardImages = [card_blur_1, card_blur_2, card_blur_3, card_blur_4,
         card_blur_5]
        arrOfSlotImageViews = [slot_1, slot_2, slot_3, slot_4, slot_5]
        arrOfBackgrounds = [bg_1, bg_2, bg_3, bg_4, bg_5]
        arrOfKeepLabels = [keep_1, keep_2, keep_3, keep_4, keep_5]
    }
    //----------------------//----------------------
    func createCardObjectsFromImages() {
        card_blur_1 = UIImage(named: "blur_1.png")
        card_blur_2 = UIImage(named: "blur_2.png")
        card_blur_3 = UIImage(named: "blur_3.png")
        card_blur_4 = UIImage(named: "blur_4.png")
        card_blur_5 = UIImage(named: "blur_4.png")
    }
    //----------------------//----------------------
    func prepareAnimations(duration d: Double,
                           repeating r: Int,
                           cards c: [UIImage]) {
        for slotAnimation in arrOfSlotImageViews {
            slotAnimation.animationDuration = d
            slotAnimation.animationRepeatCount = r
            slotAnimation.animationImages = returnRandomBlurCards(arrBlurCards: c)
        }
    }
    //----------------------//----------------------
    func returnRandomBlurCards(arrBlurCards: [UIImage]) -> [UIImage] {
        var arrToReturn = [UIImage]()
        var arrOriginal = arrBlurCards
        for _ in 0..<arrBlurCards.count {
            let randomIndex = Int(arc4random_uniform(UInt32(arrOriginal.count)))
            arrToReturn.append(arrOriginal[randomIndex])
            arrOriginal.remove(at: randomIndex)
        }
        return arrToReturn
    }
    //----------------------//----------------------
    @IBAction func play(_ sender: UIButton) {
        //---
        if chances == 0 || dealButton.alpha == 0.5 {
            return
        } else {
            chances = chances - 1
        }
        //---
        var allSelected = true
        for slotAnimation in arrOfSlotImageViews {
            if slotAnimation.layer.borderWidth != 1.0 {
                allSelected = false
                break
            }
        }
        if allSelected {
            displayRandomCards()
            return
        }
        //---
        for slotAnimation in arrOfSlotImageViews {
            if slotAnimation.layer.borderWidth != 1.0 {
                slotAnimation.startAnimating()
            }
        }
        //---
        Timer.scheduledTimer(timeInterval: 2.55,
                             target: self,
                             selector: #selector(displayRandomCards),
                             userInfo: nil,
                             repeats: false)
    }
    //----------------------//----------------------
    @objc func displayRandomCards() {
        //---
        theHand = returnRandomHand()
        //---
        let arrOfCards = createCards(theHand: theHand)
        //---
        displayCards(arrOfCards: arrOfCards)
        //---
        permissionToSelectCards = true
        //---
        prepareForNextHand()
        //---
    }
    //----------------------//----------------------
    func prepareForNextHand() {
        //---
        if chances == 0 {
            permissionToSelectCards = false
            dealButton.alpha = 0.5
            resetCards()
            createDeckOfCards()
            handToAnalyse = [(0, ""), (0, ""), (0, ""), (0, ""), (0, "")]
            chances = 2
            bet = 0
            betLabel.text = "MISE : 0"
        }
        //---
    }
```

```
//----------------------//----------------------
func displayCards(arrOfCards: [String]) {
    //---
    var counter = 0
    for slotAnimation in arrOfSlotImageViews {
        if slotAnimation.layer.borderWidth != 1 {

            if chances == 0 {
                handToAnalyse = removeEmptySlotsAndReturnArray()
                handToAnalyse.append(theHand[counter])
            }
            //---

            slotAnimation.image = UIImage(named: arrOfCards[counter])
        }
        counter = counter + 1
    }
    //---
    if chances == 0 {
        verifyHand(hand: handToAnalyse)
    }
    //---
}
//----------------------//----------------------
func removeEmptySlotsAndReturnArray() -> [(Int, String)] {
    var arrToReturn = [(Int, String)]()
    for card in handToAnalyse {
        if card != (0, "") {
            arrToReturn.append(card)
        }
    }
    return arrToReturn
}
//----------------------//----------------------
func createCards(theHand: [(Int, String)]) -> [String] {
    //---
    let card_1 = "\(theHand[0].0)\(theHand[0].1).png"
    let card_2 = "\(theHand[1].0)\(theHand[1].1).png"
    let card_3 = "\(theHand[2].0)\(theHand[2].1).png"
    let card_4 = "\(theHand[3].0)\(theHand[3].1).png"
    let card_5 = "\(theHand[4].0)\(theHand[4].1).png"
    return [card_1, card_2, card_3, card_4, card_5]
    //---
}
//----------------------//----------------------
func returnRandomHand() -> [(Int, String)] {
    //---
    var arrToReturn = [(Int, String)]()
    //---
    for _ in 1...5 {
        let randomIndex = Int(arc4random_uniform(UInt32(deckOfCards.count)))
        arrToReturn.append(deckOfCards[randomIndex])
        deckOfCards.remove(at: randomIndex)
    }
    //---
    return arrToReturn
    //---
}
//----------------------//----------------------
func verifyHand(hand: [(Int, String)]) {
    if pokerHands.royalFlush(hand: hand) {
        calculateHand(times: 250, handToDisplay: "QUINTE FLUSH ROYALE")
    } else if pokerHands.straightFlush(hand: hand) {
        calculateHand(times: 50, handToDisplay: "QUINTE FLUSH")
    } else if pokerHands.fourKind(hand: hand) {
        calculateHand(times: 25, handToDisplay: "CARRÉ")
    } else if pokerHands.fullHouse(hand: hand) {
        calculateHand(times: 9, handToDisplay: "FULL")
    } else if pokerHands.flush(hand: hand) {
        calculateHand(times: 6, handToDisplay: "COULEUR")
    } else if pokerHands.straight(hand: hand) {
        calculateHand(times: 4, handToDisplay: "QUINTE")
    } else if pokerHands.threeKind(hand: hand) {
        calculateHand(times: 3, handToDisplay: "BRELAN")
    } else if pokerHands.twoPairs(hand: hand) {
        calculateHand(times: 2, handToDisplay: "DEUX PAIRES")
    } else if pokerHands.onePair(hand: hand) {
        calculateHand(times: 1, handToDisplay: "PAIRE")
    } else {
        calculateHand(times: 0, handToDisplay: "RIEN...")
    }
}
//----------------------//----------------------
func calculateHand(times: Int, handToDisplay: String) {
    credits += (times * bet)
    tempLabel.text = handToDisplay
    creditsLabel.text = "CRÉDITS: \(credits)"
}
//----------------------//----------------------
@IBAction func cardsToHold(_ sender: UIButton) {
    //---
    if !permissionToSelectCards {
        return
    }
    //---
    if arrOfBackgrounds[sender.tag].layer.borderWidth == 0.5 {
        arrOfSlotImageViews[sender.tag].layer.borderWidth = 0.5
        arrOfBackgrounds[sender.tag].layer.borderWidth = 0.0
        arrOfBackgrounds[sender.tag].layer.backgroundColor = nil
        arrOfKeepLabels[sender.tag].isHidden = true
        //---
        manageSelectedCards(theTag: sender.tag, shouldAdd: false)
    } else {
        arrOfSlotImageViews[sender.tag].layer.borderWidth = 1.0
        arrOfBackgrounds[sender.tag].layer.borderWidth = 0.5
        arrOfBackgrounds[sender.tag].layer.borderColor = UIColor.blue.cgColor
        arrOfBackgrounds[sender.tag].layer.backgroundColor = UIColor(red: 0.0,
         green: 0.0, blue: 1.0, alpha: 0.5).cgColor
```

```
            arrOfKeepLabels[sender.tag].isHidden = false
            //---
            manageSelectedCards(theTag: sender.tag, shouldAdd: true)
        }
    }
    //----------------------//----------------------
    func manageSelectedCards(theTag: Int, shouldAdd: Bool) {
        if shouldAdd {
            handToAnalyse[theTag] = theHand[theTag]
        } else {
            handToAnalyse[theTag] = (0, "")
        }
    }
    //----------------------//----------------------
    @IBAction func betButtons(_ sender: UIButton) {
        //---
        if chances <= 1 {
            return
        }
        //---
        tempLabel.text = ""
        //---
        if sender.tag == 1000 {
            bet = credits
            betLabel.text = "MISE : \(bet)"
            credits = 0
            creditsLabel.text = "CRÉDITS : \(credits)"
            dealButton.alpha = 1.0
            resetBackOfCards()
            return
        }
        //---
        let theBet = sender.tag
        //---
        if credits >= theBet {
            bet += theBet
            credits -= theBet
            betLabel.text = "MISE : \(bet)"
            creditsLabel.text = "CRÉDITS : \(credits)"
            dealButton.alpha = 1.0
        }
        //---
        resetBackOfCards()
        //---
    }
    //----------------------//----------------------
    func resetBackOfCards() {
        for back in arrOfSlotImageViews {
            back.image = UIImage(named: "back.png")
        }
    }
    //----------------------//----------------------
    func resetCards() {
        //---
        for index in 0...4 {
            arrOfSlotImageViews[index].layer.borderWidth = 0.5
            arrOfBackgrounds[index].layer.borderWidth = 0.0
            arrOfBackgrounds[index].layer.backgroundColor = nil
            arrOfKeepLabels[index].isHidden = true
        }
        //---
        chances = 2
        //---
    }
    //----------------------//----------------------
}
//----------------------//----------------------
```