```swift
//
//  ViewController.swift
//  C-Tilt
//
//  Created by Scott Bera on 8/3/21.
//

import UIKit
import CoreMotion
import AVFoundation
import GameKit

class ViewController: UIViewController, GKGameCenterControllerDelegate {

    let motionManager = CMMotionManager()
    var audioPlayer = AVAudioPlayer()
    override var preferredScreenEdgesDeferringSystemGestures: UIRectEdge {
        return [.bottom]
    }

    var currentColor: Int = 0
    var currentScore: Int = 0
    var actionBoxColor: Int = 0
    var randomTilt: Int = 0
    var actionBoxTilt: String = ""
    var actionBoxSpeed: Double = 0.0
    var deviceTilt: String = ""
    var deviceRoll: Double = 0.0
    var devicePitch: Double = 0.0
    var bestScore: Int = 0
    var helpScreen: Bool = false
    var moreScreen: Bool = false
    var randomMusic: Int = 0
    var soundToggle: Bool = true
    var vibrateToggle: Bool = true
    var speedToggle: Int = 1
    var happyEaster: Int = 0
    var sound = Bundle.main.path(forResource: nil, ofType: nil)

    @IBOutlet weak var tealBox: UIView!
    @IBOutlet weak var yellowBox: UIView!
    @IBOutlet weak var pinkBox: UIView!
    @IBOutlet weak var greenBox: UIView!
    @IBOutlet weak var contactBar: UIView!
    @IBOutlet weak var actionBox: UIView!
    @IBOutlet weak var numberCounter: UILabel!
    @IBOutlet weak var angleBarDown: UIView!
    @IBOutlet weak var angleBarUp: UIView!
    @IBOutlet weak var angleBarLeft: UIView!
    @IBOutlet weak var angleBarRight: UIView!
    @IBOutlet weak var actionArrow: UIImageView!
    @IBOutlet weak var helpButton: UIButton!
    @IBOutlet weak var playButton: UIButton!
    @IBOutlet weak var moreButton: UIButton!
    @IBOutlet weak var instructionBoxAngle: UILabel!
    @IBOutlet weak var instructionBoxMain: UILabel!
    @IBOutlet weak var instructionBoxContact: UILabel!
    @IBOutlet weak var bestScoreLabel: UILabel!
    @IBOutlet weak var currentSong: UILabel!
    @IBOutlet weak var musicButton: UIButton!
    @IBOutlet weak var hapticsButton: UIButton!
```

```
@IBOutlet weak var speedButton: UIButton!
@IBOutlet weak var gcButton: UIButton!
@IBOutlet weak var infoButton: UIButton!
@IBOutlet weak var instructionBoxMore: UILabel!
@IBOutlet weak var giraffeImage: UIImageView!

override func viewDidLoad() {
    super.viewDidLoad()

    motionManager.deviceMotionUpdateInterval = 0.1
    motionManager.startDeviceMotionUpdates(
        to: OperationQueue.current!, withHandler: {
            (deviceMotion, error) -> Void in

            if(error == nil) {
                self.handleDeviceMotionUpdate(deviceMotion: deviceMotion!)
            } else {
                print("Critical Core Motion Error")
            }
        })

    let BestScoreDefault = UserDefaults.standard
    if (BestScoreDefault.value(forKey: "Best") != nil) {
        bestScore = BestScoreDefault.value(forKey: "Best") as! NSInteger
    }

    authPlayer()

    bestScoreLabel.text = "BEST: \(bestScore)"
    currentSong.numberOfLines = 0

    instructionBoxAngle.alpha = 0.0
    instructionBoxMain.alpha = 0.0
    instructionBoxContact.alpha = 0.0
    bestScoreLabel.alpha = 1.0
    currentSong.alpha = 0.0
    instructionBoxMore.alpha = 0.0
    giraffeImage.alpha = 0.0

    musicButton.alpha = 0.0
    hapticsButton.alpha = 0.0
    speedButton.alpha = 0.0
    gcButton.alpha = 0.0
    infoButton.alpha = 0.0

    actionBox.alpha = 0.0
    actionBox.center.x = view.center.x

    angleBarUp.layer.cornerRadius = 2
    angleBarDown.layer.cornerRadius = 2
    angleBarRight.layer.cornerRadius = 2
    angleBarLeft.layer.cornerRadius = 2

    playButton.layer.borderColor = UIColor.white.cgColor
    helpButton.layer.borderColor = UIColor.white.cgColor
    moreButton.layer.borderColor = UIColor.white.cgColor

    musicButton.layer.borderColor = UIColor.white.cgColor
    hapticsButton.layer.borderColor = UIColor.white.cgColor
    speedButton.layer.borderColor = UIColor.white.cgColor
    gcButton.layer.borderColor = UIColor.white.cgColor
```

```
        infoButton.layer.borderColor = UIColor.white.cgColor
        giraffeImage.layer.cornerRadius = 5

        tealBox.layer.borderColor = UIColor.systemTeal.cgColor
        yellowBox.layer.borderColor = UIColor.systemYellow.cgColor
        pinkBox.layer.borderColor = UIColor.systemPink.cgColor
        greenBox.layer.borderColor = UIColor.systemGreen.cgColor

        styleButton(item: playButton)
        styleButton(item: helpButton)
        styleButton(item: moreButton)

        styleButton(item: musicButton)
        styleButton(item: hapticsButton)
        styleButton(item: speedButton)
        styleButton(item: gcButton)
        styleButton(item: infoButton)

        styleButton(item: tealBox)
        styleButton(item: yellowBox)
        styleButton(item: pinkBox)
        styleButton(item: greenBox)

        setupTapTeal()
        setupTapYellow()
        setupTapPink()
        setupTapGreen()

    }

    func styleButton(item: UIView) {
        item.layer.cornerRadius = 5
        item.backgroundColor = UIColor.black
        item.layer.borderWidth = 2
    }

    func handleDeviceMotionUpdate(deviceMotion:CMDeviceMotion) {

        deviceRoll = deviceMotion.attitude.roll
        //print("ROLL = \(deviceRoll)")
        devicePitch = deviceMotion.attitude.pitch
        //print("PITCH = \(devicePitch)")

        if abs(deviceRoll) > abs(devicePitch) {
            if deviceRoll < -0.0 {
                deviceTilt = "left"
            }
            else if deviceRoll > 0.0 {
                deviceTilt = "right"
            }
            else {
                deviceTilt = "center"
            }
        }

        else if abs(deviceRoll) < abs(devicePitch) {
            if devicePitch < -0.0 {
                deviceTilt = "up"
            }
            else if devicePitch > 0.0 {
                deviceTilt = "down"
            }
            else {
                deviceTilt = "center"
            }
        }

        if deviceTilt == "left" {
            angleBarLeft.alpha = 1.0
        }
        else {
            angleBarLeft.alpha = 0.0
        }

        if deviceTilt == "right" {
            angleBarRight.alpha = 1.0
        }
        else {
            angleBarRight.alpha = 0.0
        }

        if deviceTilt == "up" {
            angleBarUp.alpha = 1.0
        }
        else {
            angleBarUp.alpha = 0.0
        }

        if deviceTilt == "down" {
            angleBarDown.alpha = 1.0
        }
        else {
            angleBarDown.alpha = 0.0
        }

        //print(actionBox.center.y)

    }

    func setupTapTeal() {
        let touchDown = UILongPressGestureRecognizer(target:self, action: #selector(tealPressed))
        touchDown.minimumPressDuration = 0
        tealBox.addGestureRecognizer(touchDown)
        }

    func setupTapYellow() {
        let touchDown = UILongPressGestureRecognizer(target:self, action: #selector(yellowPressed))
        touchDown.minimumPressDuration = 0
        yellowBox.addGestureRecognizer(touchDown)
        }

    func setupTapPink() {
        let touchDown = UILongPressGestureRecognizer(target:self, action: #selector(pinkPressed))
        touchDown.minimumPressDuration = 0
        pinkBox.addGestureRecognizer(touchDown)
        }

    func setupTapGreen() {
        let touchDown = UILongPressGestureRecognizer(target:self, action: #selector(greenPressed))
        touchDown.minimumPressDuration = 0
        greenBox.addGestureRecognizer(touchDown)
        }
```

```
@IBAction func tealPressed (recognizer: UILongPressGestureRecognizer){

    // TEAL represented by 1
    if recognizer.state == .began {
        vibrateClick()
        tealBox.backgroundColor = UIColor.systemTeal
        yellowBox.backgroundColor = UIColor.black
        pinkBox.backgroundColor = UIColor.black
        greenBox.backgroundColor = UIColor.black
        contactBar.backgroundColor = UIColor.systemTeal
        giraffeImage.alpha = 0.0
        currentColor = 1
        happyEaster += 1
        print("tealPressed")

    }

    else if recognizer.state == .ended || recognizer.state == .cancelled{
        tealBox.backgroundColor = UIColor.black
        happyEaster -= 1

        if currentColor == 1 {
            contactBar.backgroundColor = UIColor.white
            currentColor = 0
        }

        if helpScreen {
            resetMenu(playDirection: 1, helpDirection: -1, moreDirection: 1)
        }

        if moreScreen {
            resetMenu(playDirection: 1, helpDirection: 1, moreDirection: -1)
        }
    }

}

@IBAction func yellowPressed (recognizer: UILongPressGestureRecognizer){

    // YELLOW represented by 2

    if recognizer.state == .began {
        vibrateClick()
        yellowBox.backgroundColor = UIColor.systemYellow
        tealBox.backgroundColor = UIColor.black
        pinkBox.backgroundColor = UIColor.black
        greenBox.backgroundColor = UIColor.black
        contactBar.backgroundColor = UIColor.systemYellow
        currentColor = 2
        happyEaster += 1
        print("yellowPressed")

    }

    else if recognizer.state == .ended || recognizer.state == .cancelled{
        yellowBox.backgroundColor = UIColor.black
        happyEaster -= 1

        if currentColor == 2 {
            contactBar.backgroundColor = UIColor.white
```

```
                currentColor = 0
            }

            if helpScreen {
                resetMenu(playDirection: 1, helpDirection: -1, moreDirection: 1)
            }

            if moreScreen {
                resetMenu(playDirection: 1, helpDirection: 1, moreDirection: -1)
            }
        }

    }

    @IBAction func pinkPressed (recognizer: UILongPressGestureRecognizer){

        // PINK represented by 3

        if recognizer.state == .began {
            vibrateClick()
            pinkBox.backgroundColor = UIColor.systemPink
            tealBox.backgroundColor = UIColor.black
            yellowBox.backgroundColor = UIColor.black
            greenBox.backgroundColor = UIColor.black
            contactBar.backgroundColor = UIColor.systemPink
            currentColor = 3
            happyEaster += 1
            print("pinkPressed")

        }

        else if recognizer.state == .ended || recognizer.state == .cancelled{
            pinkBox.backgroundColor = UIColor.black
            happyEaster -= 1

            if currentColor == 3 {
                contactBar.backgroundColor = UIColor.white
                currentColor = 0
            }

            if helpScreen {
                resetMenu(playDirection: 1, helpDirection: -1, moreDirection: 1)
            }

            if moreScreen {
                resetMenu(playDirection: 1, helpDirection: 1, moreDirection: -1)
            }
        }

    }

    @IBAction func greenPressed (recognizer: UILongPressGestureRecognizer){

        // GREEN represented by 4

        if recognizer.state == .began {
            vibrateClick()
            greenBox.backgroundColor = UIColor.systemGreen
            tealBox.backgroundColor = UIColor.black
            yellowBox.backgroundColor = UIColor.black
            pinkBox.backgroundColor = UIColor.black
            contactBar.backgroundColor = UIColor.systemGreen
            currentColor = 4
            happyEaster += 1
            print("greenPressed")

        }

        else if recognizer.state == .ended || recognizer.state == .cancelled{
            greenBox.backgroundColor = UIColor.black
            happyEaster -= 1

            if currentColor == 4 {
                contactBar.backgroundColor = UIColor.white
                currentColor = 0
            }

            if helpScreen {
                resetMenu(playDirection: 1, helpDirection: -1, moreDirection: 1)
            }

            if moreScreen {
                resetMenu(playDirection: 1, helpDirection: 1, moreDirection: -1)
            }
        }

    }

    @IBAction func playPressed(_ sender: Any) {

        print("PLAY BUTTON PRESSED")

        currentScore = 0
        numberCounter.text = "00\(String(currentScore))"

        if speedToggle == 1 {
            actionBoxSpeed = 5.0
        }

        else if speedToggle == 2 {
            actionBoxSpeed = 1.776321832470721
        }

        else if speedToggle == 3 {
            actionBoxSpeed = 1.3119364218173402
        }

        vibrateClick()

        if happyEaster == 4 {
            giraffeImage.alpha = 1.0
            print("giraf")
        }

        UIView.animate(
            withDuration: 0.5,
            animations: {
                self.playButton.transform = self.playButton.transform.translatedBy(x: 250, y: 0)
                self.helpButton.transform = self.helpButton.transform.translatedBy(x: -250, y: 0)
                self.moreButton.transform = self.moreButton.transform.translatedBy(x: -250, y: 0)

                self.moreButton.alpha = 0.0
                self.playButton.alpha = 0.0
                self.helpButton.alpha = 0.0
                self.bestScoreLabel.alpha = 0.0
            },
            completion: {finished in
                if self.soundToggle {
                    self.chooseMusic()
                }
                else {
                    self.currentSong.text = "MUSIC OFF"
                    UIView.animate(
                        withDuration: 0.5,
                        animations: {
                            self.currentSong.alpha = 1.5
                        },
                        completion: {finished in
                            UIView.animate(
                                withDuration: 2.0,
                                animations: {
                                    self.currentSong.alpha = 0.0
                                },
                                completion: {finished in self.resetActionBox()}
                            )
                        }
                    )
                }
            }
        )

    }

    func chooseMusic() {

        if bestScore < 5 {
            randomMusic = 4
        }
        else {
            if randomMusic != 100 {
                randomMusic = Int.random(in: 1...11)
            }
        }

        switch randomMusic {
        case 1:
             sound = Bundle.main.path(forResource: "music_zapsplat_game_music_action_retro_8_bit_repeating_016", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: Action Retro \nBY: Alan McKinney"

        case 2:
             sound = Bundle.main.path(forResource: "music_zapsplat_game_music_action_fun_funky_electro_disco_023", ofType:
".mp3")
            currentSong.text = "NOW PLAYING: Action Funky \nBY: Alan McKinney"

        case 3:
             sound = Bundle.main.path(forResource:
"music_zapsplat_game_music_action_fast_euro_house_pumping_fun_arcade_rave_024", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: Arcade Rave \nBY: Alan McKinney"

        case 4:
             sound = Bundle.main.path(forResource: "music_zapsplat_astro_race", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: Astro Race \nBY: Alan McKinney"
        case 5:
             sound = Bundle.main.path(forResource: "ES_U & Me - SLCT", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: U & Me \nBY: SLCT"

        case 6:
             sound = Bundle.main.path(forResource: "ES_A Vivid Dream by an 8 Bit Machine - Rymdklang Soundtracks", ofType:
".mp3")
            currentSong.text = "NOW PLAYING: A Vivid Dream by an 8 bit Machine \nBY: Rymdklang Soundtracks"

//        case 7:
//             sound = Bundle.main.path(forResource: "a-big-adventure by fassounds Artlist", ofType: ".mp3") //HAS WATERMARK
//            currentSong.text = "NOW PLAYING: A Big Adventure \nBY: Game Up!"

        case 7:
             sound = Bundle.main.path(forResource: "471_full_l-a-nights_0160", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: LA Nights \nBY: Full Frontal Audio"

        case 8:
             sound = Bundle.main.path(forResource: "462_full_no-vertical-limit_0164", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: No Vertical Limit \nBY: J. Fontaine"

        case 9:
             sound = Bundle.main.path(forResource:
"music_zapsplat_game_music_arcade_electro_repeating_retro_arp_electro_drums_serious_012", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: Electro Drums \nBY: Alan McKinney"

        case 10:
             sound = Bundle.main.path(forResource:
"music_zapsplat_game_music_action_racing_fast_paced_electronic_synth_challenging_009", ofType: ".mp3")
            currentSong.text = "NOW PLAYING: Racing Synth \nBY: Alan McKinney"

        case 11:
             sound = Bundle.main.path(forResource: "music_zapsplat_game_music_action_fast_paced_elelctro_sonic_011", ofType:
".mp3")
            currentSong.text = "NOW PLAYING: Electro Sonic \nBY: Alan McKinney"

        case 100:
            currentSong.text = ":)"

        default: print ("MUSIC RANDOMIZATION ERROR")
        }

        if sound != nil {
            do {
                audioPlayer = try AVAudioPlayer(contentsOf: URL(fileURLWithPath: sound!))
            }
            catch {
                print("MUSIC PLAYBACK ERROR")
            }

            audioPlayer.numberOfLoops = -1
            audioPlayer.play()

            print("Now Playing: \(String(describing: sound?.description))")
        }

        UIView.animate(
            withDuration: 0.5,
            animations: {
                self.currentSong.alpha = 1.5
            },
            completion: {finished in
                UIView.animate(
                    withDuration: 2.0,
                    animations: {
                        self.currentSong.alpha = 0.0
                    },
                    completion: {finished in self.resetActionBox()}
                )
            }
        )

    }

    @IBAction func helpPressed(_ sender: Any) {

        print("HELP PRESSED")

        vibrateClick()

        UIView.animate(
            withDuration: 0.5,
            animations: {
                self.playButton.transform = self.playButton.transform.translatedBy(x: -250, y: 0)
                self.helpButton.transform = self.helpButton.transform.translatedBy(x: 250, y: 0)
                self.moreButton.transform = self.moreButton.transform.translatedBy(x: -250, y: 0)

                self.playButton.alpha = 0.0
                self.helpButton.alpha = 0.0
                self.moreButton.alpha = 0.0
                self.bestScoreLabel.alpha = 0.0
                self.instructionBoxAngle.alpha = 1.0
                self.instructionBoxMain.alpha = 1.0
                self.instructionBoxContact.alpha = 1.0

            }
        )

//        if happyEaster == 4 {
//            randomMusic = 100
//            print("kile sotin")
//        }

        helpScreen = true

    }
    @IBAction func morePressed(_ sender: Any) {

        print("MORE PRESSED")

        vibrateClick()

        UIView.animate(
            withDuration: 0.5,
            animations: {
                self.playButton.transform = self.playButton.transform.translatedBy(x: -250, y: 0)
                self.helpButton.transform = self.helpButton.transform.translatedBy(x: -250, y: 0)
                self.moreButton.transform = self.moreButton.transform.translatedBy(x: 250, y: 0)

                self.playButton.alpha = 0.0
                self.helpButton.alpha = 0.0
                self.moreButton.alpha = 0.0
                self.bestScoreLabel.alpha = 0.0

                self.musicButton.alpha = 1.0
                self.hapticsButton.alpha = 1.0
                self.speedButton.alpha = 1.0
                self.gcButton.alpha = 1.0
                self.infoButton.alpha = 1.0
                self.instructionBoxMore.alpha = 1.0

            }
        )

        moreScreen = true
    }

    func resetMenu(playDirection: Int, helpDirection: Int, moreDirection: Int) {

        helpScreen = false
        moreScreen = false

        UIView.animate(
            withDuration: 0.5,
            animations: {
                self.playButton.transform = self.playButton.transform.translatedBy(x: CGFloat(250 * playDirection), y: 0)
                self.helpButton.transform = self.helpButton.transform.translatedBy(x: CGFloat(250 * helpDirection), y: 0)
                self.moreButton.transform = self.moreButton.transform.translatedBy(x: CGFloat(250 * moreDirection), y: 0)

                self.playButton.alpha = 1.0
                self.helpButton.alpha = 1.0
                self.moreButton.alpha = 1.0
                self.bestScoreLabel.alpha = 1.0
                self.instructionBoxAngle.alpha = 0.0
                self.instructionBoxMain.alpha = 0.0
                self.instructionBoxContact.alpha = 0.0
                self.actionBox.alpha = 0.0

                self.musicButton.alpha = 0.0
                self.hapticsButton.alpha = 0.0
                self.speedButton.alpha = 0.0
                self.gcButton.alpha = 0.0
                self.infoButton.alpha = 0.0
                self.instructionBoxMore.alpha = 0.0

                }
            )

        print("MENU RESET")

    }

    func sendActionBox() {

        print("ACTION BOX SENT")
        //print("number counter = \(self.numberCounter.center.y)")
        //print("contact bar = \(self.contactBar.center.y)")
        //print("action box = \(self.actionBox.center.y)")


        actionBoxColor = Int.random(in: 1...4)

        if actionBoxColor == 1 {
            actionBox.backgroundColor = UIColor.systemTeal
        }
        if actionBoxColor == 2 {
            actionBox.backgroundColor = UIColor.systemYellow
        }
        if actionBoxColor == 3 {
            actionBox.backgroundColor = UIColor.systemPink
        }
        if actionBoxColor == 4 {
            actionBox.backgroundColor = UIColor.systemGreen
        }

        print("Action Box Color: \(actionBoxColor)")

        randomTilt = Int.random(in: 1...4)

        if randomTilt == 1 {
            actionBoxTilt = "left"
            actionArrow.transform = actionArrow.transform.rotated(by: .pi/2)
        }
        if randomTilt == 2 {
            actionBoxTilt = "right"
            actionArrow.transform = actionArrow.transform.rotated(by: -.pi/2)
        }
        if randomTilt == 3 {
            actionBoxTilt = "up"
            actionArrow.transform = actionArrow.transform.rotated(by: .pi)
        }
        if randomTilt == 4 {
            actionBoxTilt = "down"
            actionArrow.transform = actionArrow.transform.rotated(by: 0)
        }

        print("Action Box Tilt: \(actionBoxTilt)")

        actionBox.alpha = 1.0

        UIView.animate(
            withDuration: actionBoxSpeed,
            animations: {
                //if self.actionBoxSpeed == 5.0 {
                    //self.actionBox.center.y = self.contactBar.center.y - 95
               // }
               // else {
                    self.actionBox.center.y = self.contactBar.center.y - 25

                //}
            },
            completion: {finished in self.checkMatch()}
        )

    }

    func checkMatch() {
        print("MATCH CHECKED")

        //print(actionBox.center.y) //TEMP
        //print(contactBar.center.y) //TEMP

        if currentColor == actionBoxColor && deviceTilt == actionBoxTilt {
            currentScore += 1
```

```swift
            print("MATCH CORRECT, score = \(currentScore)")
            vibrateMatch()
            resetActionBox()
        }

        else {

//            currentScore += 1
//            print("MATCH CORRECT, score = \(currentScore)")
//            vibrateMatch()
//            resetActionBox()

            print("GAME OVER, score = \(currentScore)")
            bestScoreLabel.text = "BEST: \(bestScore)"
            saveGC()
            audioPlayer.stop()
            audioPlayer.currentTime = 0
            randomMusic = 0
            giraffeImage.alpha = 0.0

            if currentColor != actionBoxColor {
                actionBox.backgroundColor = UIColor.white
            }
            if deviceTilt != actionBoxTilt {
                actionArrow.tintColor = UIColor.white
            }

            resetMenu(playDirection: -1, helpDirection: 1, moreDirection: 1)

            }
    }

    func resetActionBox() {
        print("-> ACTION BOX RESET")

        actionBox.alpha = 0.0
        self.actionBox.center.y = self.numberCounter.center.y - 150
        actionArrow.transform = CGAffineTransform.identity
        actionArrow.tintColor = UIColor.black
        updateNumberCounter()
        }

    func updateNumberCounter() {
        print("NUMBER COUNTER UPDATED")

        if currentScore >= 10 {
            if currentScore >= 100 {
                numberCounter.text = String(currentScore)
            }
            else {
                numberCounter.text = "0\(String(currentScore))"
            }
        }
        else {
            numberCounter.text = "00\(String(currentScore))"
        }

        if currentScore > bestScore {
            bestScore = currentScore

            let BestScoreDefault = UserDefaults.standard
        BestScoreDefault.setValue(bestScore, forKey: "Best")
        BestScoreDefault.synchronize()

        print("BEST SCORE UPDATED, best score = \(bestScore)")
    }

    updateActionBoxSpeed()
}

func updateActionBoxSpeed() {

    if speedToggle == 1 {
        if currentScore < 15 {
            actionBoxSpeed -= actionBoxSpeed / 15
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else if currentScore < 30 {
            actionBoxSpeed -= actionBoxSpeed / 50
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else if currentScore < 100 {
            actionBoxSpeed -= actionBoxSpeed / 250
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else {
            actionBoxSpeed -= actionBoxSpeed / 1000
            print("ACTION BOX SPEED UPDATED PAST 100, speed = \(actionBoxSpeed)")
        }
    }

    else if speedToggle == 2 {
         if currentScore < 15 {
            actionBoxSpeed -= actionBoxSpeed / 50
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else if currentScore < 85 {
            actionBoxSpeed -= actionBoxSpeed / 250
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else {
            actionBoxSpeed -= actionBoxSpeed / 1000
            print("ACTION BOX SPEED UPDATED PAST 100, speed = \(actionBoxSpeed)")
        }
    }

    else if speedToggle == 3 {
         if currentScore < 70 {
            actionBoxSpeed -= actionBoxSpeed / 250
            print("ACTION BOX SPEED UPDATED, speed = \(actionBoxSpeed)")
        }
        else {
            actionBoxSpeed -= actionBoxSpeed / 1000
            print("ACTION BOX SPEED UPDATED PAST 100, speed = \(actionBoxSpeed)")
        }
    }

    sendActionBox()
}

func vibrateClick() {
    if vibrateToggle {
        let clickFeedback = UISelectionFeedbackGenerator()
        clickFeedback.prepare()
        clickFeedback.selectionChanged()
    }
}

func vibrateMatch() {
    if vibrateToggle {
        let matchFeedback = UIImpactFeedbackGenerator()
        matchFeedback.prepare()
        matchFeedback.impactOccurred()
    }
}

@IBAction func musicPressed(_ sender: Any) {

    vibrateClick()

    if soundToggle {
        soundToggle = false
        musicButton.setTitle("MUSIC: OFF", for: .normal)
        print("music off")
    }
    else {
        soundToggle = true
        musicButton.setTitle("MUSIC: ON", for: .normal)
        print("music on")
    }
}

@IBAction func vibratePressed(_ sender: Any) {
    if vibrateToggle {
        vibrateToggle = false
        hapticsButton.setTitle("VIBRATE: OFF", for: .normal)
        print("vibrate off")
    }
    else {
        vibrateToggle = true
        hapticsButton.setTitle("VIBRATE: ON", for: .normal)
        vibrateClick()
        print("vibrate on")
    }
}

@IBAction func speedPressed(_ sender: Any) {

    vibrateClick()

    if speedToggle == 1 {
        speedToggle = 2
        speedButton.setTitle("SPEED: 002", for: .normal)
        print("speed = 2")
    }
    else if speedToggle == 2 {
        speedToggle = 3
        speedButton.setTitle("SPEED: 003", for: .normal)
        print("speed = 3")
    }
    else if speedToggle == 3 {
        speedToggle = 1
        speedButton.setTitle("SPEED: 001", for: .normal)
```

```
            print("speed = 1")
        }
    }

    @IBAction func infoPressed(_ sender: Any) {
        vibrateClick()
        print("info pressed")
        UIApplication.shared.open(NSURL(string: "https://linktr.ee/scottbera")! as URL)
    }

    @IBAction func gcPressed(_ sender: Any) {
        vibrateClick()
        print("gamecenter pressed")

        let viewController = self.view.window?.rootViewController
        let gcvc = GKGameCenterViewController()
        gcvc.gameCenterDelegate = self
        viewController?.present(gcvc, animated: true, completion: nil)

    }

    func authPlayer() {
        let localPlayer = GKLocalPlayer.local
        localPlayer.authenticateHandler = {
            (view, error) in
            if view != nil {
                self.present(view!, animated: true, completion: nil)
            }
            else {
                print("GC Player Authenticated: \(GKLocalPlayer.local.isAuthenticated)")
            }
        }
    }

    func saveGC() {

        if GKLocalPlayer.local.isAuthenticated {

            GKLeaderboard.submitScore(currentScore, context: 0, player: GKLocalPlayer.local, leaderboardIDs: ["overall"],
completionHandler: {_ in })

            if speedToggle == 1 {
                GKLeaderboard.submitScore(currentScore, context: 0, player: GKLocalPlayer.local, leaderboardIDs: ["HISCORES"],
completionHandler: {_ in })
            }

            else if speedToggle == 2 {
                GKLeaderboard.submitScore(currentScore, context: 0, player: GKLocalPlayer.local, leaderboardIDs: ["all_time"],
completionHandler: {_ in })
            }

            else if speedToggle == 3 {
                GKLeaderboard.submitScore(currentScore, context: 0, player: GKLocalPlayer.local, leaderboardIDs: ["speed_3"],
completionHandler: {_ in })
            }
        }
    }

    func gameCenterViewControllerDidFinish(_ gameCenterViewController: GKGameCenterViewController) {
        gameCenterViewController.dismiss(animated: true, completion: nil)
    }
```

```
}
```