SCIENCES SORBONNE UNIVERSITÉ

«Accelerometer»

Fabrice.Kordon@lip6.fr

LIP6

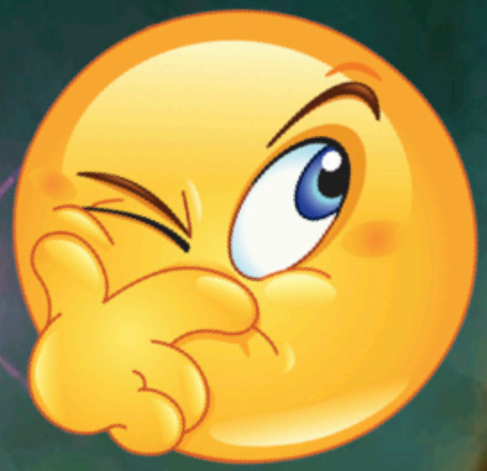
Goal of the example

Play with the sensors

- Accelerometer
- Gyroscope
- Magnetometer

What to do?

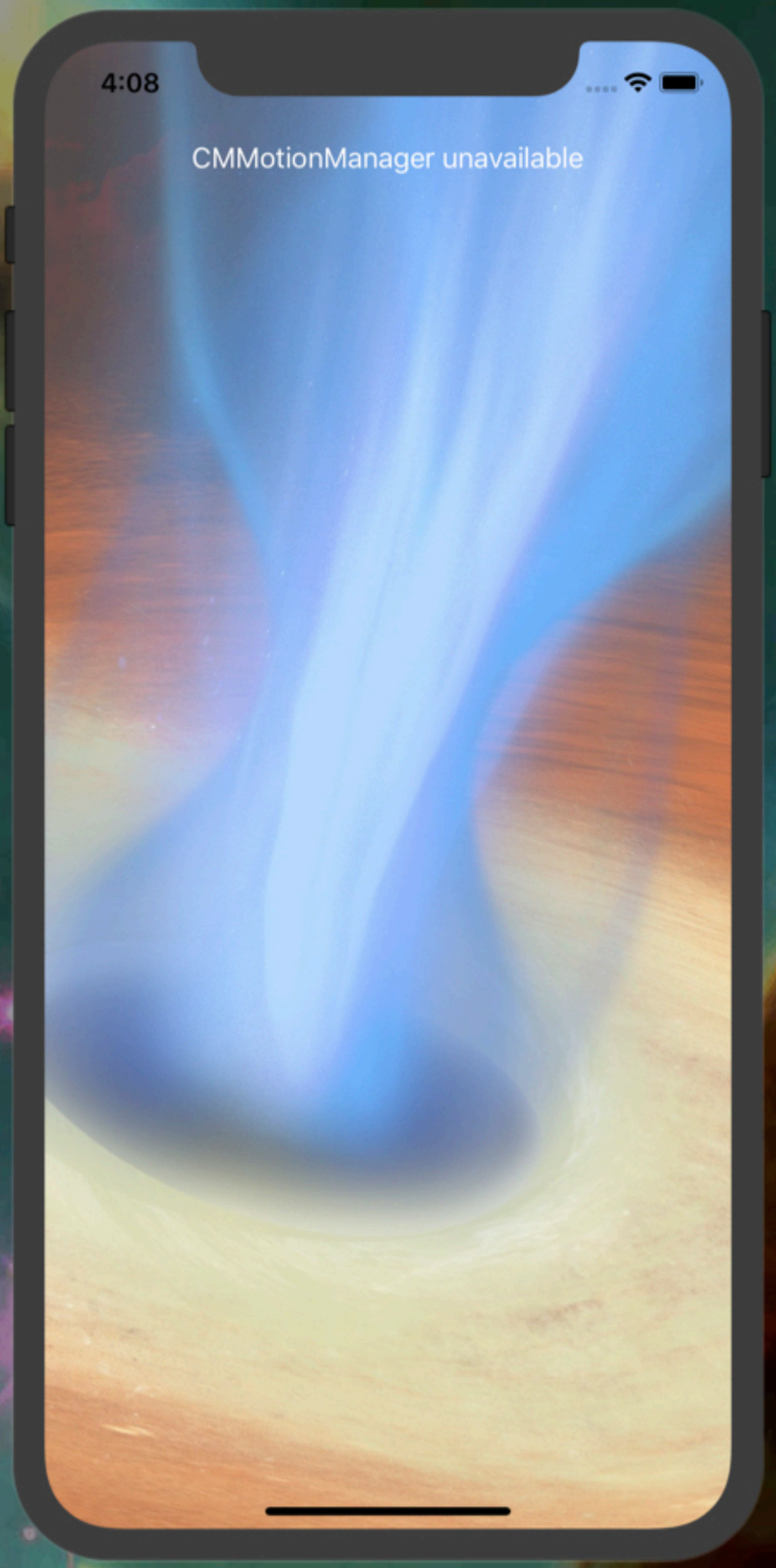
- Display values
- Stabilise an image
  - **Inspired from http://nshipster.com/cmdevicemotion**



Demo (iPhone)

Demo (iPod touch)

Demo (predefined, simulator)



ViewController


```swift
//
//  ViewController.swift
//  Accelerometer
//
//  Created by Fabrice Kordon on 02/11/2018.
//  Copyright © 2018 Sorbonne Université. All rights reserved.
//

import UIKit

class ViewController: UIViewController {

    private let v = MyView(frame: UIScreen.main.bounds)

    override func viewDidLoad() {
        super.viewDidLoad()
        self.view = v
    }

    override func viewWillTransition(to size: CGSize,
                        with coordinator: UIViewControllerTransitionCoordinator) {
        v.displayInSize(size)
    }
}
```

MyView


```
import UIKit
import CoreMotion // do not forget

class MyView: UIView {
    private let l1 = UILabel()
    private let l2 = UILabel()
    private let l3 = UILabel()
    private let frequency = UISegmentedControl(items: ["1Hz", "10Hz", "100Hz"])
    private let sensor = UISegmentedControl()
    private var timer : Timer?
    private var interval : TimeInterval = 1.0
    private let b = UIButton(type: .system)
    private let cmMngr = CMMotionManager()
    private let earth = UIImageView(image: UIImage(named: "earth"))
    private let backg = UIImageView(image: UIImage(named: "backgimg"))
```

MyView


```
override init(frame: CGRect) {
    super.init(frame: frame)
    self.backgroundColor = UIColor.white
    backg.alpha = 0.7
    self.addSubview(backg)
    if cmMngr.isDeviceMotionAvailable {
        l1.text = "---"
        l1.textColor = UIColor.black
        l1.textAlignment = .center
        l2.text = "---"
        l2.textAlignment = .center
        l2.textColor = UIColor.black
        l3.text = "---"
        l3.textAlignment = .center
        l3.textColor = UIColor.black
        frequency.tintColor = UIColor.black
        frequency.selectedSegmentIndex = 0
        frequency.addTarget(self, action: #selector(updateFrequency),
                            for: .valueChanged)
        if cmMngr.isAccelerometerAvailable {
            sensor.insertSegment(withTitle: "Accelerometer",
                                 at: sensor.numberOfSegments,
                                 animated: false)
        }
        if cmMngr.isGyroAvailable {
            sensor.insertSegment(withTitle: "Gyroscope",
                                 at: sensor.numberOfSegments,
                                 animated: false)
        }
```

MyView


```
        if cmMngr.isMagnetometerAvailable {
            sensor.insertSegment(withTitle: "Magnetometer",
                                 at: sensor.numberOfSegments,
                                 animated: false)
        }
        sensor.tintColor = UIColor.black
        sensor.selectedSegmentIndex = 0
        if UIDevice.current.userInterfaceIdiom == .phone {
            earth.frame = CGRect(x: 0.0, y: 0.0,
                                 width: 200.0, height: 200.0)
        } else {
            earth.frame = CGRect(x: 0.0, y: 0.0,
                                 width: 350.0, height: 350.0)
        }
```

```
            b.setTitle("Start", for: .normal)
            b.tintColor = UIColor.black
            b.addTarget(self, action: #selector(startStop), for: .touchDown)
            self.addSubview(l2)
            self.addSubview(l3)
            self.addSubview(frequency)
            self.addSubview(sensor)
            self.addSubview(b)
            self.addSubview(earth)
        } else {
            l1.text = "CMMotionManager unavailable"
            l1.textColor = UIColor.white
            l1.textAlignment = .center
        }
        self.addSubview(l1)
        self.displayInSize(UIScreen.main.bounds.size)
    }
```

MyView


```
required init(coder aDecoder: NSCoder) {
   fatalError("init(coder:) has not been implemented")
}

func displayInSize(_ size: CGSize) {
   var top = 20;
   var left = 20
   if UIDevice.current.userInterfaceIdiom == .phone &&
       size.height >= 812 {
       top = 30
   } else if UIDevice.current.userInterfaceIdiom == .phone &&
       size.width > size.height {
       top = 0
       if size.width >= 812 {
           left = 40
       }
   }
   backg.center = CGPoint(x: size.width / 2, y: size.height / 2)
   if size.width > size.height &&
       UIDevice.current.userInterfaceIdiom == .phone {
       l1.frame = CGRect(x: left, y: top + 30, width: Int(size.width / 2) - 2 * left, height: 20)
       l2.frame = CGRect(x: left, y: top + 60, width: Int(size.width / 2) - 2 * left, height: 20)
       l3.frame = CGRect(x: left, y: top + 90, width: Int(size.width / 2) - 2 * left, height: 20)
       sensor.frame = CGRect(x: left, y: top + 130, width: Int(size.width) / 2 - 2 * left, height: 30)
       frequency.frame = CGRect(x: left, y: top + 180, width: Int(size.width / 2) - 2 * left, height: 30)
       b.frame  = CGRect(x: left, y: top + 230, width: Int(size.width / 2) - 2 * left, height: 20)
       earth.center = CGPoint(x: size.width / 4 * 3, y: size.height / 2)

   } else {
       l1.frame = CGRect(x: left, y: top + 30, width: Int(size.width) - 2 * left, height: 20)
       l2.frame = CGRect(x: left, y: top + 60, width: Int(size.width) - 2 * left, height: 20)
       l3.frame = CGRect(x: left, y: top + 90, width: Int(size.width) - 2 * left, height: 20)
       sensor.frame = CGRect(x: left, y: top + 130, width: Int(size.width) - 2 * left, height: 30)
       frequency.frame = CGRect(x: left, y: top + 180, width: Int(size.width) - 2 * left, height: 30)
       b.frame  = CGRect(x: left, y: top + 230, width: Int(size.width) - 2 * left, height: 20)
       earth.center = CGPoint(x: size.width / 2, y: size.height / 3 * 2)
   }
}
```

MyView


```
@objc func updateFrequency () {
    switch frequency.selectedSegmentIndex {
        case 0: interval = 1.0
        case 1: interval = 0.1
        case 2: interval = 0.01
        default: () // never reached
    }
    if b.title(for: .normal) == "Stop" {
        self.resetTimer()
    }
    // for technique 1, do this:
    // cmMngr.deviceMotionUpdateInterval = interval
}
```

MyView


```
@objc func startStop () {
    if b.title(for: .normal) == "Stop" {
        b.setTitle("Démarrer", for: .normal)
        cmMngr.stopDeviceMotionUpdates()
        cmMngr.stopAccelerometerUpdates()
        cmMngr.stopGyroUpdates()
        timer?.invalidate()
        timer = nil
        l1.text = "---"
        l2.text = "---"
        l3.text = "---"
    } else {
        b.setTitle("Stop", for: .normal)
        cmMngr.startDeviceMotionUpdates()
        cmMngr.startMagnetometerUpdates()
        cmMngr.startGyroUpdates()
        self.resetTimer()
    }
}
```

MyView


```
// Using the second technique presented in the slides

func resetTimer () {
    if timer !== nil {
        timer!.invalidate()
    }
    timer = Timer.scheduledTimer(timeInterval: interval,
                                 target: self,
                                 selector: #selector(updateView),
                                 userInfo: nil, repeats: true)
}
```

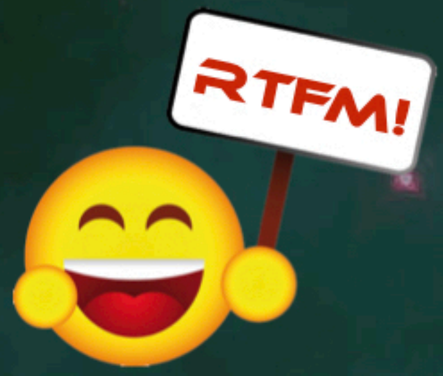
MyView


```
@objc func updateView () {
    if cmMngr.deviceMotion !== nil { // to avoid crash...
        if cmMngr.isAccelerometerAvailable { // have the earth rotate
            var delta = Double.pi
            if UIDevice.current.orientation == .landscapeLeft {
                delta = -Double.pi / 2
            } else if UIDevice.current.orientation == .landscapeRight {
                delta = Double.pi / 2
            }
            let rotation = atan2((cmMngr.deviceMotion?.gravity.x)!,
                                 (cmMngr.deviceMotion?.gravity.y)!) - delta
            earth.transform = CGAffineTransform(rotationAngle: CGFloat(rotation))
            } if sensor.titleForSegment(at: sensor.selectedSegmentIndex) == "Accelerometer" {
                l1.text = String(format: "x = %4.4f", (cmMngr.deviceMotion?.gravity.x)!)
                l2.text = String(format: "y = %4.4f", (cmMngr.deviceMotion?.gravity.y)!)
                l3.text = String(format: "z = %4.4f", (cmMngr.deviceMotion?.gravity.z)!)
            } if sensor.titleForSegment(at: sensor.selectedSegmentIndex) == "Gyroscope" {
                l1.text = String(format: "roll = %4.4f",
                                 (cmMngr.deviceMotion?.attitude.roll)!)
                l2.text = String(format: "pitch = %4.4f",
                                 (cmMngr.deviceMotion?.attitude.pitch)!)
                l3.text = String(format: "yaw = %4.4f",
                                 (cmMngr.deviceMotion?.attitude.yaw)!)
            } if sensor.titleForSegment(at: sensor.selectedSegmentIndex) == "Magnetometer" {
                l1.text = String(format: "cal.x = %4.4f",
                                  (cmMngr.magnetometerData?.magneticField.x)!)
                l2.text = String(format: "cal.y = %4.4f",
                                 (cmMngr.magnetometerData?.magneticField.y)!)
                l3.text = String(format: "cal.z = %4.4f",
                                 (cmMngr.magnetometerData?.magneticField.z)!)
            }
        }
    }
}
```

As a conclusion…

We presented technique 2

- The other is easy to retrieve
  - **Good exercice for you**
- Be aware of
  - **Using OperationQueue**
  - **Updating the sampling frequency (deviceMotionUpdateInterval)**



Be cautious with energy

- Remember Instrument can help you
  - **A way to monitor energy**