SCIENCES SORBONNE UNIVERSITÉ

«SysInfo»

Fabrice.Kordon@lip6.fr

LIP6

Goal of the example

Query UIDevice

- Activate standard notifications
- Display related information

Demo (iPhone)

ViewController

Sake of simplicity…

Code located in a ViewController

ViewController


```
import UIKit

class ViewController: UIViewController {

    private let orientation = UILabel()
    private let batteryState = UILabel()
    private let batteryLevel = UILabel()
    private let proximity = UILabel()
    private let device = UILabel()
    private let multitask = UILabel()
    private let OS = UILabel()
private let b = UIButton(type: .system)
private let myDevice = UIDevice.current
private let nCenter = NotificationCenter.default
private let progress = UIProgressView()

override var preferredStatusBarStyle : UIStatusBarStyle {
    return .lightContent
}
```

ViewController


```
override func viewDidLoad() {
    super.viewDidLoad()
    self.view = UIView(frame: UIScreen.main.bounds)
    self.view.backgroundColor = .black
    orientation.text = "Orientation :"
    orientation.textColor = .white
    self.view.addSubview(orientation)
    batteryState.text = "Battery state :"
    batteryState.textColor = .white
    self.view.addSubview(batteryState)
    batteryLevel.text = "Battery level :"
    batteryLevel.textColor = .white
    self.view.addSubview(batteryLevel)
    proximity.text = "Proximity sensor :"
    proximity.textColor = .white
    self.view.addSubview(proximity)
    device.text = "Device name : \(myDevice.name)"
    device.textColor = .white
    self.view.addSubview(device)
     if myDevice.isMultitaskingSupported {
         multitask.text = "Multitask : yes"
     } else {
         multitask.text = "Multitask : no"
     }
     multitask.textColor = .white
     self.view.addSubview(multitask)
     OS.text = "Operating System : \(myDevice.systemName) \(myDevice.systemVersion)"
     OS.textColor = .white
     self.view.addSubview(OS)
     b.setTitle("Start", for: .normal)
     b.tintColor = .yellow
     b.addTarget(self, action: #selector(requestData), for: .touchDown)
     self.view.addSubview(b)
     self.drawInFormat(UIScreen.main.bounds.size)
}
```

ViewController


```
func drawInFormat(_ s : CGSize) {
    orientation.frame = CGRect(x: 20, y: 50, width: s.width - 40, height: 30)
    batteryState.frame = CGRect(x: 20, y: 90, width: s.width - 40, height: 30)
    batteryLevel.frame = CGRect(x: 20, y: 130, width: s.width - 40, height: 30)
    proximity.frame = CGRect(x: 20, y: 170, width: s.width - 40, height: 30)
    device.frame = CGRect(x: 20, y: 210, width: s.width - 40, height: 30)
    OS.frame = CGRect(x: 20, y: 250, width: s.width - 40, height: 30)
    multitask.frame = CGRect(x: 20, y: 290, width: s.width - 40, height: 30)
    b.frame = CGRect(x: 20, y: 330, width: s.width - 40, height: 30)
    progress.frame = CGRect(x: 20, y: 370, width: s.width - 40, height: 30)
}
```

ViewController


```
@objc func requestData(sender : UIButton) {
    if b.titleLabel!.text == "Start" {
        b.setTitle("Stop", for: .normal)
        myDevice.beginGeneratingDeviceOrientationNotifications()
        myDevice.isBatteryMonitoringEnabled = true
        myDevice.isProximityMonitoringEnabled = true
        nCenter.addObserver(self, selector: #selector(notificationUpdate),
                            name: UIDevice.orientationDidChangeNotification,
                            object: nil)
        nCenter.addObserver(self, selector: #selector(notificationUpdate),
                            name:  UIDevice.batteryLevelDidChangeNotification,
                            object: nil)
        nCenter.addObserver(self, selector: #selector(notificationUpdate),
                            name:  UIDevice.batteryStateDidChangeNotification,
                            object: nil)
        nCenter.addObserver(self, selector: #selector(notificationUpdate),
                            name:  UIDevice.proximityStateDidChangeNotification,
                            object: nil)
        // Capture current state
        updateOrientation()
        updateBatteryLevel()
        updateBatteryState()
        updateProximity()
    } else {
        b.setTitle("Start", for: .normal)
        myDevice.endGeneratingDeviceOrientationNotifications()
        myDevice.isBatteryMonitoringEnabled = false
        myDevice.isProximityMonitoringEnabled = false
        nCenter.removeObserver(self)
    }
}
```

ViewController


```
@objc func notificationUpdate(n : Notification) {
    switch n.name {
    case UIDevice.orientationDidChangeNotification:
        updateOrientation()
    case UIDevice.batteryLevelDidChangeNotification:
        updateBatteryLevel()
    case UIDevice.batteryStateDidChangeNotification:
        updateBatteryState()
    case UIDevice.proximityStateDidChangeNotification:
        updateProximity()
    default:
        print("should not happend ;-)")
    }
}
```

ViewController


```
@objc func updateOrientation() {
    switch myDevice.orientation {
    case .portrait:
        orientation.text = "Orientation : portrait"
    case .portraitUpsideDown:
        orientation.text = "Orientation : upside down"
    case .landscapeLeft:
        orientation.text = "Orientation : landscape right"
    case .landscapeRight:
        orientation.text = "Orientation : landscape left"
    case .faceUp:
        orientation.text = "Orientation : face up"
    case .faceDown:
        orientation.text = "Orientation : face down"
    default:
        orientation.text = "Orientation : unknown"
    }
}

func updateBatteryState() {
    switch myDevice.batteryState {
    case .full:
        batteryState.text = "Battery state : full"
    case .unplugged:
        batteryState.text = "Battery state : unplugged"
    case .charging:
        batteryState.text = "Battery state : charging"
    default:
        batteryState.text = "Battery state : unknown"
    }
}
```

ViewController


```
func updateProximity() {
    if myDevice.isProximityMonitoringEnabled {
        if myDevice.proximityState {
            proximity.text = "Proximity sensor : enabled"
        } else {
            proximity.text = "Proximity sensor : disabled"
        }
    } else {
        proximity.text = "Proximity sensor : not available"
    }
}
```

```
    func updateBatteryLevel() {
        if myDevice.batteryLevel == -1 {
            batteryLevel.text = "Battery level : unsupported"
        } else {
            batteryLevel.text = "Battery level : \(myDevice.batteryLevel * 100)%"
        }
    }
}
```

As a conclusion…

You know it now!!!

- And you even played a bit with notifications

- You have no excuse screwing device management !