SCIENCES SORBONNE UNIVERSITÉ

«Watch&Phone»

Fabrice.Kordon@lip6.fr

LIP6

Goal of the example

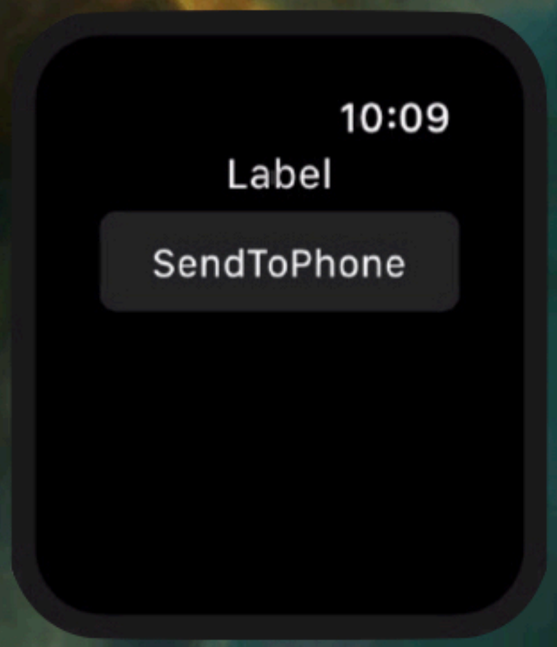
Let phone and watch exchange messages

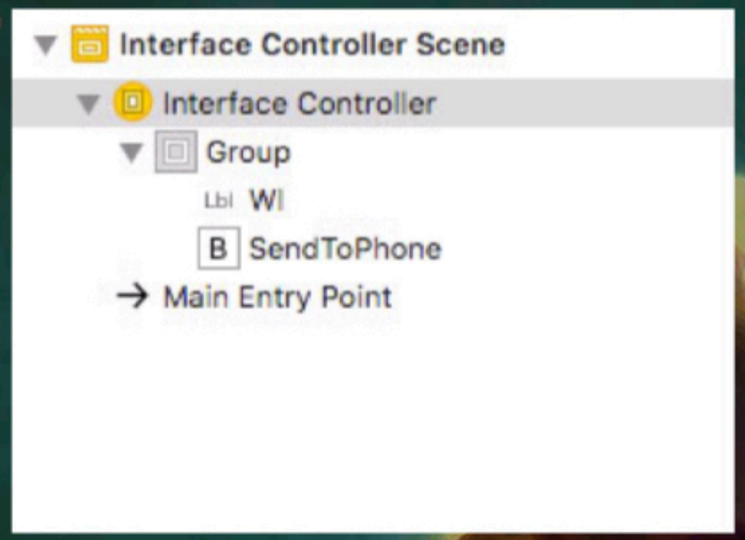
- A simple string
  - **Sent on one side**
  - **Displayed on the other side**
- Check for potential problems
  - **Device that cannot be paired**

Demo (unpaired device)

Demo (paired device)

Storyboard


10:09
Label
SendToPhone


Interface Controller Scene
Interface Controller
Group
Lbl WI
B SendToPhone
Main Entry Point

A bit about conventions

- One message type only
  - A string
    - Key «msg»
    - Value, any string

ViewController

Sake of simplicity…

Code located in a ViewController

ViewController


```
import UIKit
import WatchConnectivity

class ViewController: UIViewController, WCSessionDelegate {

    private let l = UILabel()
    private let b = UIButton(type: .system)
    private let i = UIImageView(image: UIImage(named: "applewatch"))

    private let session = WCSession.default;

    private var count = 1
```

ViewController


```
override func viewDidLoad() {
    super.viewDidLoad()
    // Let's be simple, no orientation supported
    let s = UIScreen.main.bounds.size
    self.view = UIView()
    self.view.backgroundColor = .white
    self.view.addSubview(l)
    l.textAlignment = .center
    self.view.addSubview(b)
    b.setTitle("SendToWatch", for: .normal)
    b.addTarget(self, action: #selector(sendToWatch), for: .touchDown)
    self.view.addSubview(i)
    i.isHidden = true
    i.center = CGPoint(x: s.width / 2, y: s.height / 2)
    l.frame = CGRect(x: 20, y: 70, width: s.width - 40, height: 20)
    b.frame = CGRect(x: 50, y: 100, width: s.width - 100, height: 20)
    if WCSession.isSupported() {
        session.delegate = self
        session.activate()
    } else {
        let a = UIAlertController(title: "Problem",
                                  message: "No WCSession supported",
                                  preferredStyle: .alert)
        a.addAction(UIAlertAction(title: "Ok",
                                  style: .default,
                                  handler: nil))
        self.present(a, animated: true, completion: nil)
    }
}
```

ViewController


```
@objc func sendToWatch() {
    if session.isReachable {
        session.sendMessage(["msg":"Hello Watch \(count)"],
                            replyHandler: nil, errorHandler: nil)
        count += 1
    } else {
        let a = UIAlertController(title: "Problem",
                                  message: "Your Watch is unreachable",
                                  preferredStyle: .alert)
        a.addAction(UIAlertAction(title: "Ok",
                                  style: .default,
                                  handler: nil))
        self.present(a, animated: true, completion: nil)
    }
}
```

ViewController


```
// WCSessionDelegate protocol

func session(_ session: WCSession,
             activationDidCompleteWith activationState: WCSessionActivationState,
             error: Error?) {
    DispatchQueue.main.async{
        if error != nil {
            let a = UIAlertController(title: "Problem",
                                      message: error!.localizedDescription,
                                      preferredStyle: .alert)
            a.addAction(UIAlertAction(title: "Ok",
                                      style: .default,
                                      handler: nil))
            self.present(a, animated: true, completion: nil)
        } else {
            switch activationState {
            case .notActivated :
                print ("activationDidCompleteWith (notActivated)")
            case .inactive :
                print ("activationDidCompleteWith (inactive)")
            case .activated :
                print ("activationDidCompleteWith (activated)")
                if session.isPaired && session.isReachable {
                    self.i.isHidden = false
                }
            }
        }
    }
}
```

ViewController


```
func session(_ session: WCSession,
             didReceiveMessage message: [String : Any]) {
    DispatchQueue.main.async{
        let s = message["msg"] as? String
        if s != nil {
            self.l.text = s
        } else {
            let a = UIAlertController(title: "Problem",
                                      message: "Message could not be decoded",
                                      preferredStyle: .alert)
            a.addAction(UIAlertAction(title: "Ok",
                                      style: .default,
                                      handler: nil))
            self.present(a, animated: true, completion: nil)
        }
    }
}

func sessionReachabilityDidChange(_ session: WCSession) {
     DispatchQueue.main.async{
        if session.isPaired && session.isReachable {
            self.i.isHidden = false
        } else {
            self.i.isHidden = true
        }
    }
}
```

ViewController


```
func sessionDidBecomeInactive(_ session: WCSession) {
    DispatchQueue.main.async{
        self.i.isHidden = true
    }
}

func sessionDidDeactivate(_ session: WCSession) {
    print("sessionDidDeactivate")
    DispatchQueue.main.async{
        self.i.isHidden = true
    }
}
```

InterfaceController


```swift
import WatchKit
import Foundation
import WatchConnectivity

class InterfaceController: WKInterfaceController, WCSessionDelegate {

    @IBOutlet weak var wl: WKInterfaceLabel!

    private let session = WCSession.default

    private var count = 1

    override func awake(withContext context: Any?) {
        super.awake(withContext: context)
        // Configure interface objects here.
        if WCSession.isSupported() {
            session.delegate = self
            session.activate()
        }
    }

    override func willActivate() {
        super.willActivate()
    }

    override func didDeactivate() {
        super.didDeactivate()
    }
```

InterfaceController


```
@IBAction func send2Phone() {
    session.sendMessage(["msg" : "hello from Watch \(count)"],
                        replyHandler: nil,
                        errorHandler: nil)
    count += 1
}

// WCSessionDelegate protocol

func session(_ session: WCSession,
             activationDidCompleteWith activationState: WCSessionActivationState,
             error: Error?) {
    if error != nil {
        wl.setText(error!.localizedDescription)
    }
}

func session(_ session: WCSession,
             didReceiveMessage message: [String : Any]) {
    let s = message["msg"] as? String
    if s != nil {
        wl.setText(s)
    } else {
        wl.setText("Problem!!!")
    }
}
}
```

As a conclusion…

Is'n't it easy?

- Now you can add…
  - The WatchApp counterpart to your App
  - Let this counterpart exchange data with your App

Have fun with this feature!