IOT LESSONS LEARNED

HUGUES BERNET-ROLLANDE

@ROMPELSTILCHEN

GITHUB.COM/HUGUESBR

CHIEF SOFTWARE OFFICER @ WIRED BEAUTY

TOPIC

- BLE vs Bluetooth
- IoT vs Connected Object
- Not BLE tutorial

UNIT TESTS

- Debugging IoT is hard
- Data transformation layer (bytes -> value)
- Models (aggregated values)

DATA TRANSFORMATION

```
extension Data {
    func value<T>() -> T {
        return withUnsafeBytes { (ptr: UnsafePointer<T>) -> T in
            return ptr.pointee
        }
    }
}

class test: XCTestCase {
    func testInt16() {
        let bytes: [UInt8] = [0xFC, 0xFF] // big endian
        let data = Data(bytes: bytes)
        let v: Int16 = data.value() // reverse inference :)
        print(v == -4)
    }
}

extension Integer {
    public var data: Data {
        var v = self
        return Data(buffer: UnsafeBufferPointer(start: &v, count: 1))
    }
}

class IntegerTests: XCTestCase {
    func testUInt32() {
        let bytes: [UInt8] = [0x01, 0x01, 0x01, 0x03] // big endian
        let data = Data(bytes: bytes)
        XCTAssertEqual(UInt32(50_397_441).data, data)
    }
}
```

UNIT TEST YOUR MODELS

```
// 0xABCDEFGH -> "ABCDEFGH"
public struct DeviceSerial {
    public let serialString: String

    init(data: Data) throws {
        // check data length
        // reverse string (little
endian)
        // decode as hex
        self.serialString = …
    }
}}
```

```
public protocol DataRepresentable: Equatable {
    var data: Data { get }
}

public func == <T: DataRepresentable>(lhs: T, rhs: T) -> Bool {
    return lhs.data == rhs.data
}

extension DeviceSerial: DataRepresentable {
    public var data: Data {
        return …
    }
}

extension DeviceSerial {
    public init?(serialString: String) {
        // check len
        // check characters
        self.serialString = serialString
    }
}

class test: XCTestCase {
    func testValidSerial() {
        let data = Data(…)
        let a = DeviceSerial(serialString: "ABCD")
        let b = DeviceSerial(data: data)
        XCTAssertEqual(a, b)
    }
}
```

IOS SIMULATOR VS BLE

- Simulator doesn’t support BLE
- Use Wrapper & Protocols
- Stream fake Object (and data)

DEVICE MANAGER

```
protocol DeviceInterfaceProtocol {
    func getBattery(completion: (result: Result<BatteryLevel>) ->
Void)
}

public protocol DeviceManagerProtocol {
    associatedtype DeviceInterface
    var central: CBCentralManager? { get }
    var device: DeviceInterface? { get }
    var state: DeviceManagerState { get }
    var paired: Bool { get }
    init(central: CBCentralManager)
    func pair(block: (String) -> Bool)
    func unpair()
}

class DeviceManager<DeviceInterface: DeviceInterfaceProtocol >:
DeviceManagerProtocol {
  // some share default implementation
}

class DeviceWrapper {
    #if ( (arch(i386) || arch(x86_64)) && os(iOS) )
    static var sharedManager: DeviceManager = {
        return FakeDeviceManager()
    }()
    #else
    static var sharedManager: DeviceManager = {
        return DeviceManager<DeviceInterface>()
    }()
    #endif
}
```

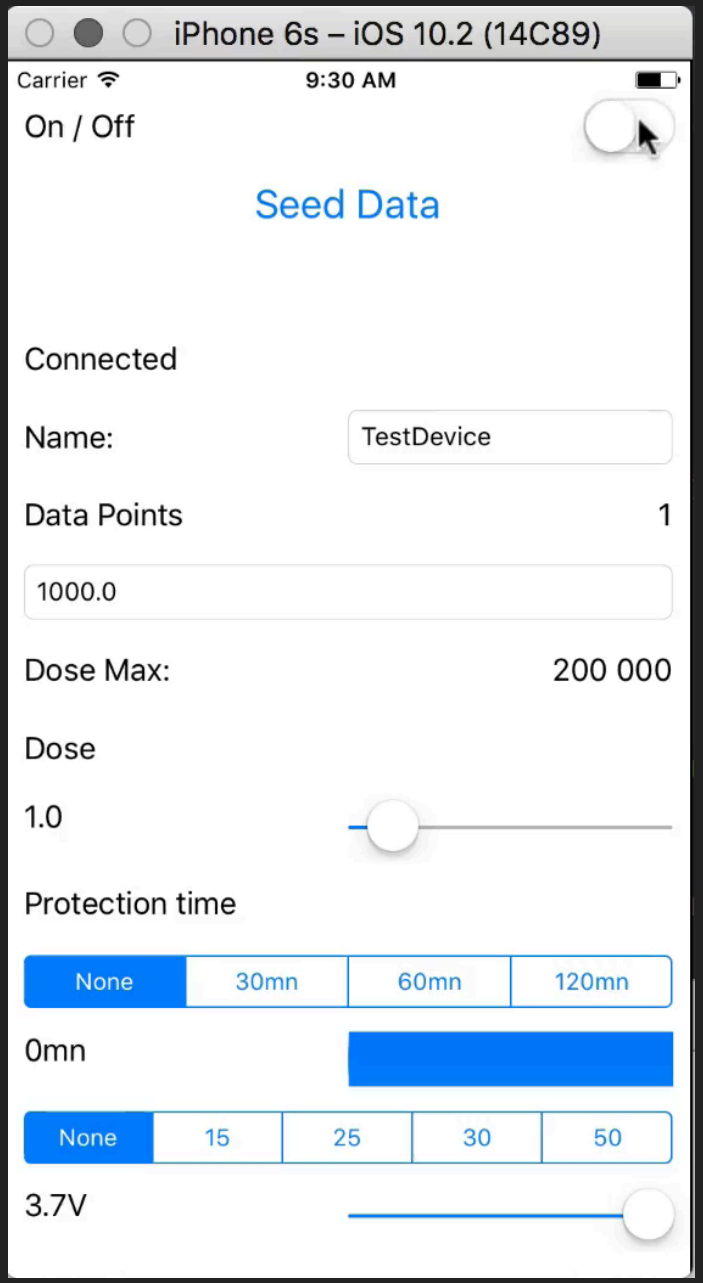
```
class FakeDeviceInterface:
DeviceInterfaceProtocol {
    func getBattery(completion: (result:
Result<BatteryLevel>) -> Void) {
        let level = BatteryLevel(voltage:
3.7)
        completion(.success(level))
    }
}

class FakeDeviceManager:
DeviceManager<FakeDeviceInterface> {
    override var state: DeviceManagerState
{
        get { return .Connected }
        set {}
    }

    override var paired: Bool {
        get { return true }
        set {}
    }
}
```

DEVICE VS APP

- Device will be slower to develop than App
- Develop a Device Simulator
- macOS Playground support BLE



ACCESS DATA INDIRECTLY

- Fetch from DB
- Ease Unit Tests
- Simulated mode

TOOLS

- Light Blue
- Apple Bluetooth Explorer
- (macOS) Playground
- Console (BTServer)

SUMMARY

- Simulate as much as you can
- Apply same principles for BLE than Server
- Unit Tests (again!)

A GOOD BLE APP IS LIKE A GOOD CAKE: IT HAS LAYER

Hugues Bernet-Rollande

@rompelstilchen

LEARN MORE ABOUT BLE

- Core Bluetooth Programming Guide
- WWDC (CoreBluetooth 101)
- Zero to BLE

https://www.cloudcity.io/blog/2015/06/11/zero-to-ble-on-ios-part-one/

https://developer.apple.com/ - Core Bluetooth Programming Guide

https://developer.apple.com/videos/play/wwdc2012/703/

THANK YOU!

SLIDES AVAILABLE ON SPEAKER DECK: HTTP://BIT.LY/2LV3ISK

HUGUES BERNET-ROLLANDE

@ROMPELSTILCHEN

GITHUB.COM/HUGUESBR

CSO ENGINEER @ WIRED BEAUTY