SCIENCES SORBONNE UNIVERSITÉ

«CarPlayHello»

Fabrice.Kordon@lip6.fr

LIP6

Goal of the example

A «Hello World» for CarPlay

- In a Navigation app mode
- Discuss deployment issues

Activate CarPlay

Difficult

- Requires a specific authorisation from Apple
  - **To get adapted profiles for deployment on a device**
  - **Probably prosecution issues**

Activation possible on the Xcode simulator

- Trick provided by Sander van Tulden
  - **He also inspired this «hello world»**
  - **Step 1: activate HomeKit (in the capabilities)**
  - **Step 2: update you key in the entitlements with com.apple.developer.carplay-maps**
- On the simulator, activate CarPlay as external display
  - **From the hardware menu**

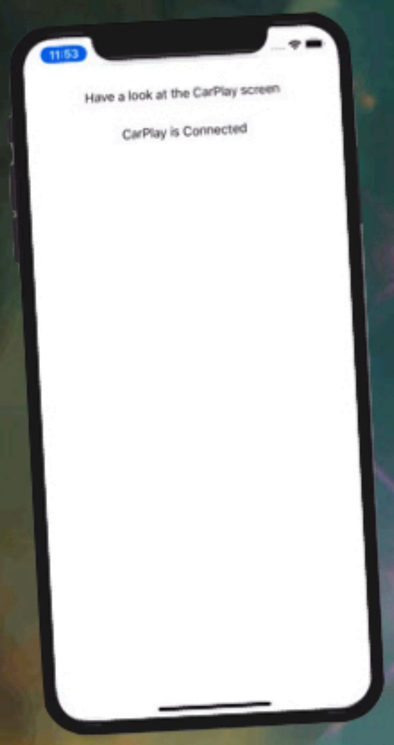
Demo (simulator)

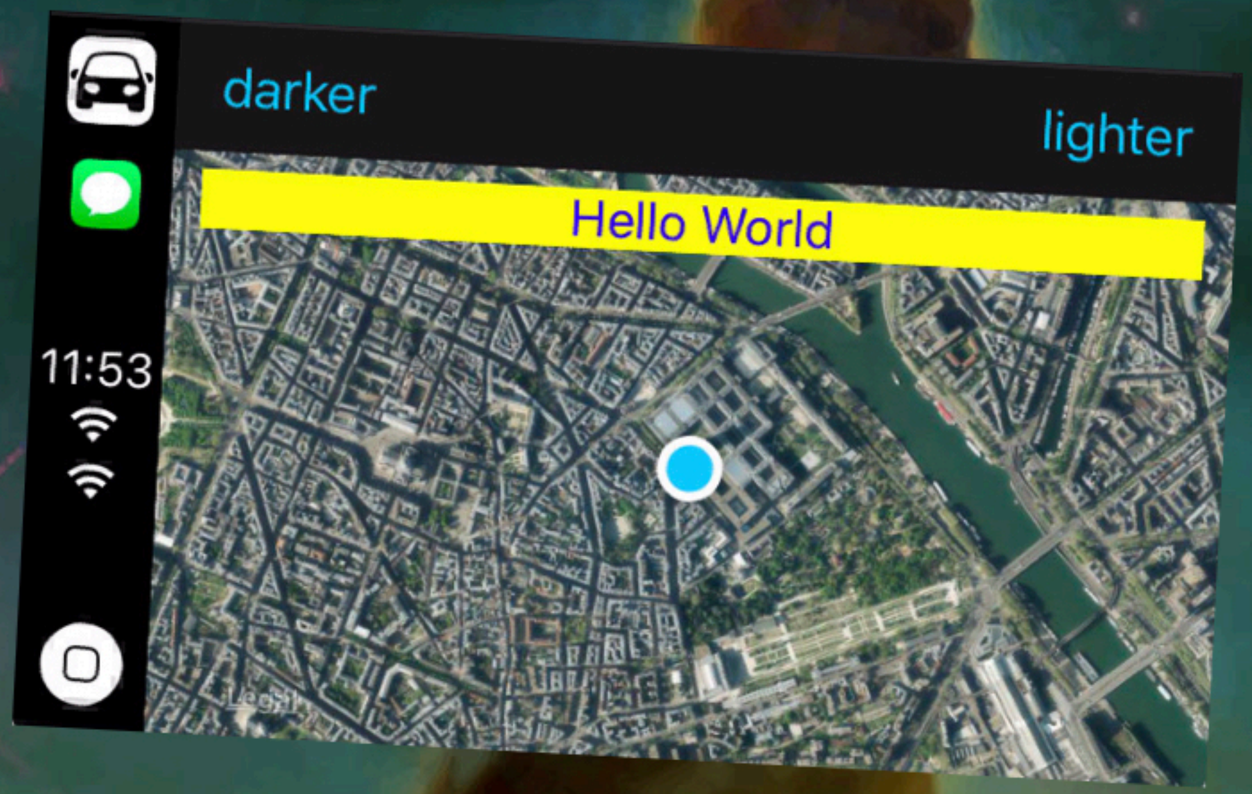
Architecture

AppDelegate

ViewController

CarPlayViewController





AppDelegate


```
import UIKit
import CarPlay

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate, CPApplicationDelegate,
CPMapTemplateDelegate {

    private var carWindow: CPWindow?
    private var interfaceController: CPInterfaceController?
    private var mapTemplate: CPMapTemplate?

enum BarButtonType: String {
    case darker = "darker"
    case lighter = "lighter"
}
```

AppDelegate


```swift
// service function to create bar buttons
private func myBarButtons(_ type: BarButtonType) -> CPBarButton {
    let barButton = CPBarButton(type: .text) { (button) in
        switch(type) {
        case .darker:
            // Dismiss the map panning interface
            self.carWindow?.rootViewController?.view.alpha = 0.5
        case .lighter:
            // Enable the map panning interface and set the dismiss button
            self.carWindow?.rootViewController?.view.alpha = 1
        }
        barButton.title = type.rawValue // Set title based on type
        return barButton
}

// service function to create a map template
func myTemplate() -> CPMapTemplate {
    // Create the default CPMapTemplate objcet (you may subclass this at your leasure
    let mapTemplate = CPMapTemplate()
    // Create the different CPBarButtons
    let darkbutton = myBarButtons(.darker)
    mapTemplate.leadingNavigationBarButtons = [darkbutton]
    let lightButton = myBarButtons(.lighter)
    mapTemplate.trailingNavigationBarButtons = [lightButton]
    // Always show the NavigationBar
    mapTemplate.automaticallyHidesNavigationBar = false
    return mapTemplate
}
```

AppDelegate


```swift
// CPApplicationDelegate

func application(_ application: UIApplication,
                 didConnectCarInterfaceController
                    interfaceController: CPInterfaceController,
                 to window: CPWindow) {
    let vc = self.window?.rootViewController as? ViewController
    vc?.displayCPConnected()
    // retrieve infromation passed to the call-backed method
    self.interfaceController = interfaceController
    self.carWindow = window
    // Creation of a CPMapTemplate (for a map)
    let mapTemplate = myTemplate()
    mapTemplate.mapDelegate = self
    self.mapTemplate = mapTemplate
    // Setting the toot of the CarPlay interface controller
    interfaceController.setRootTemplate(mapTemplate, animated: true)
    // associate your rootViewController to the window!
    window.rootViewController = CarPlayViewController()
}

func application(_ application: UIApplication,
                 didDisconnectCarInterfaceController
                    interfaceController: CPInterfaceController,
                 from window: CPWindow) {
    let vc = self.window?.rootViewController as? ViewController
    vc?.displayCPDisconnected()
}
```

AppDelegate


```
// From UIApplicationDelegate (unimplemented in this example)

var window: UIWindow?


func application(_ application: UIApplication,
                 didFinishLaunchingWithOptions launchOptions:
                   [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    // Override point for customization after application launch.
    return true
}

    func applicationWillResignActive(_ application: UIApplication) { }

    func applicationDidEnterBackground(_ application: UIApplication) { }

    func applicationWillEnterForeground(_ application: UIApplication) { }

    func applicationDidBecomeActive(_ application: UIApplication) { }

    func applicationWillTerminate(_ application: UIApplication) { }
}
```

ViewController


```swift
import UIKit

class ViewController: UIViewController {

    let l1 = UILabel()
    let l2 = UILabel()

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        self.view = UIView()
        self.view.backgroundColor = .white
        let s = UIScreen.main.bounds.size
        l1.textAlignment = .center
        l1.text = "Have a look at the CarPlay screen"
        l1.frame = CGRect(x: 10, y: 70, width: s.width - 20, height: 20)
        self.view.addSubview(l1)
        l2.textAlignment = .center
        l2.frame = CGRect(x: 10, y: 120, width: s.width - 20, height: 20)
        self.view.addSubview(l2)
    }

    func displayCPDisconnected() {
        l2.text = "CarPlay is disconnected"
    }

    func displayCPConnected() {
        l2.text = "CarPlay is Connected"
    }
}
```

CarPlayViewController


```swift
import UIKit
import CarPlay
import CoreLocation

class CarPlayViewController: UIViewController, CLLocationManagerDelegate {

    private let m = MKMapView()
    private let l = UILabel()
    private let cm = CLLocationManager()

    override func viewDidLoad() {
        super.viewDidLoad()
        self.view = m
        l.textAlignment = .center
        l.textColor = .blue
        l.text = "Hello World"
        m.addSubview(l)
        m.mapType = .satellite // not recommended in CarPlay;-)

        cm.distanceFilter = 1.0 // precision = 1m
        cm.requestWhenInUseAuthorization()
        cm.delegate = self
        cm.requestLocation() // Just once for the display
}
```

CarPlayViewController


```
override func viewWillAppear(_ animated: Bool) {
    let s = m.bounds.size
    l.frame = CGRect(x: 10 , y: 50,
                     width: s.width - 20, height: 20)
    l.backgroundColor = .yellow
}

// CLLocationManagerDelegate protocol

func locationManager(_ manager: CLLocationManager,
                     didUpdateLocations locations: [CLLocation]) {
    // Map update
    let span = MKCoordinateSpan(latitudeDelta: 0.015, longitudeDelta: 0.015)
    let region = MKCoordinateRegion(center: (manager.location?.coordinate)!,
                                    span: span)
    m.setRegion(region, animated: true)
    m.showsUserLocation = true
}

func locationManager(_ manager: CLLocationManager,
                     didFailWithError error: Error) {
    l.text = l.text!+" (location failed)"
}
}
```

As a conclusion…

Do not forget the info.plist

- To activate geolocation…

You now have «a rough idea»

- How to proceed
- More time with the FM may be required

On Android?

- AndroidAuto
  - **Probably similar problems**