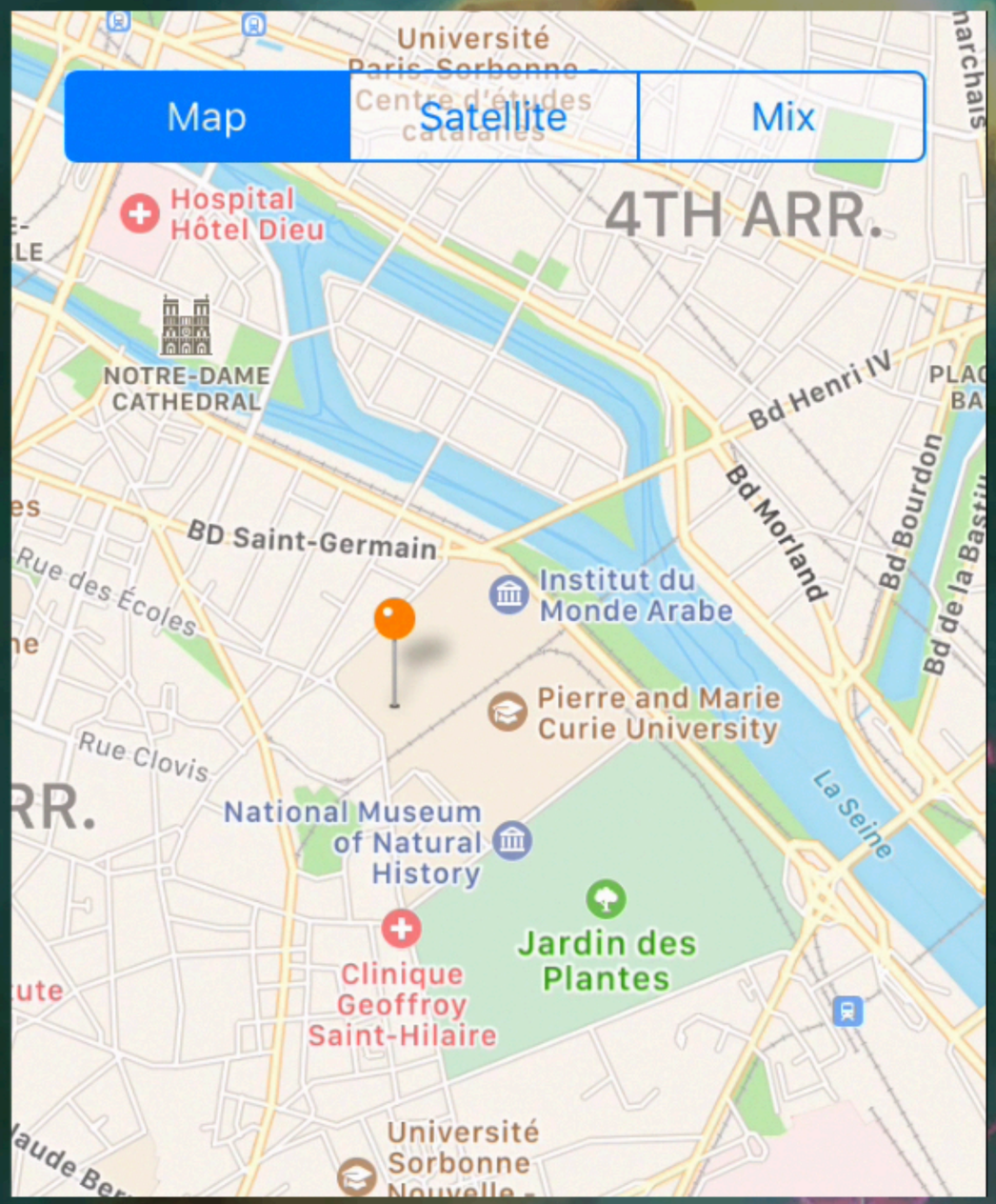
SCIENCES SORBONNE UNIVERSITÉ

Handling various types of maps

Fabrice.Kordon@lip6.fr

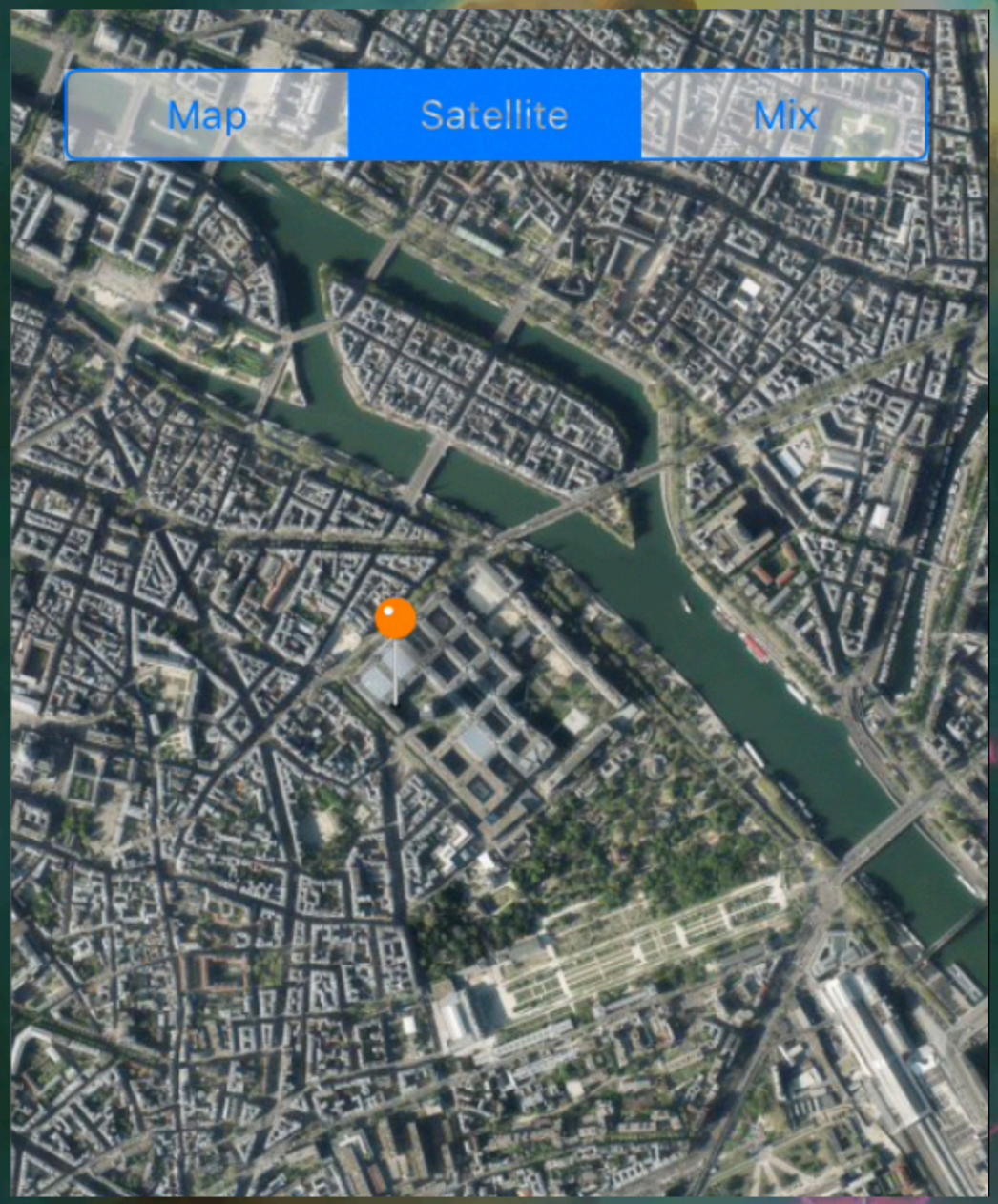
LIP6

What types of maps?



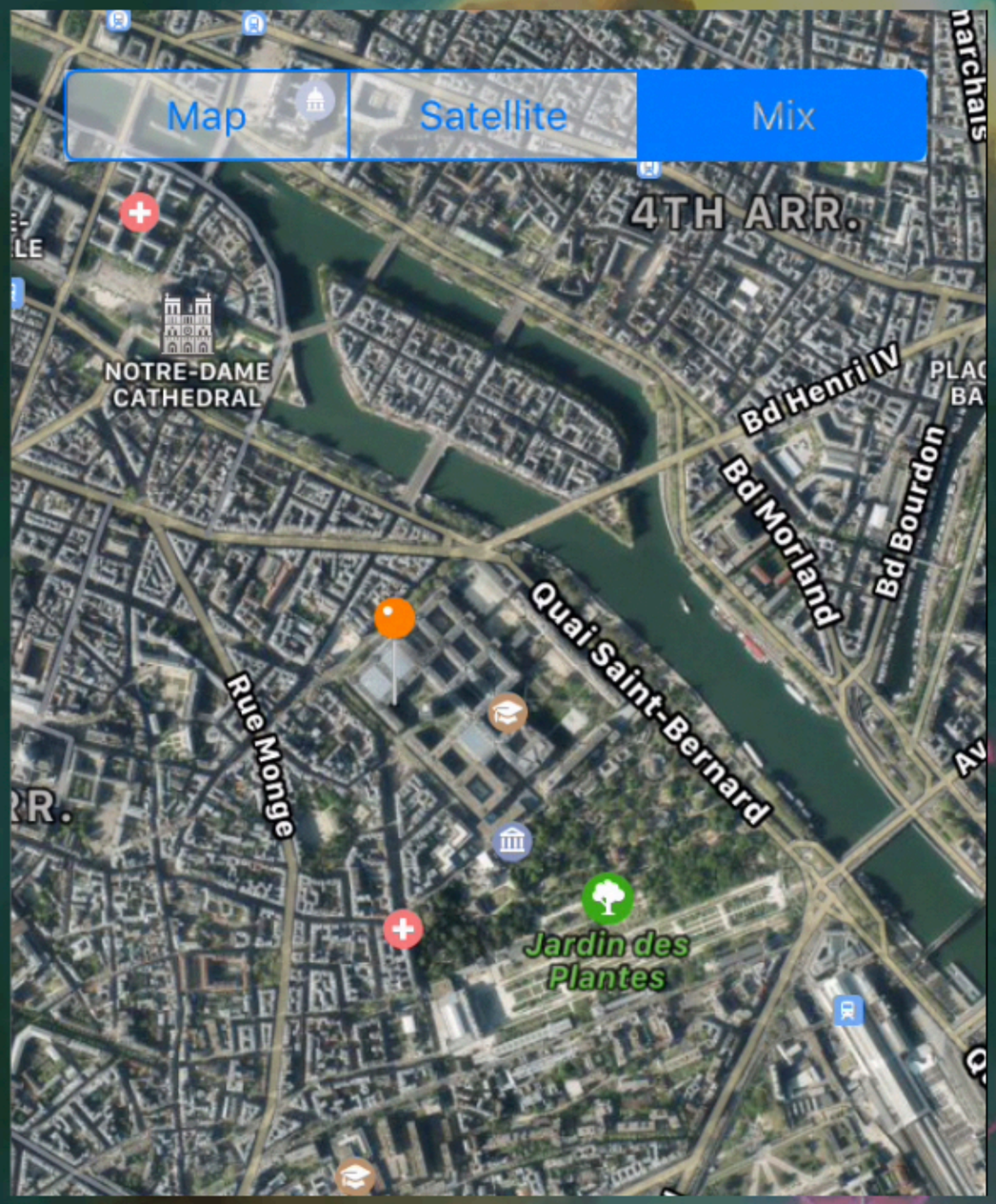
Map

What types of maps?



- Map
- Satellite

What types of maps?



- Map
- Satellite
- Hybrid

Changing the type of a map

- **Easy!!!**
  - Attribute mapType
    - **Of type MKMapType**
- **The three main values of MKMapType**
  - Swift
    - **standard, satellite, hybrid**
  - Objective-C
    - **MKMapTypeStandard, MKMapTypeSatellite, MKMapTypeHybrid**

Demo

About sources

- **AnAnnotation**
  - Does not change
- **ViewController**
  - Does not change
- **MyView**
  - Some changes…

MyView (updated)


```
import UIKit
import CoreLocation
import MapKit
class MyView: UIView, CLLocationManagerDelegate, MKMapViewDelegate {
        private let find = UIButton(type: .system)
        private let add = UIButton(type: .contactAdd)
        private let location = UITextView()
        private let lattitude = UILabel()
        private let longitude = UILabel()
        private let crtTime = UILabel()
        private let CLmngr = CLLocationManager()
        private let map = MKMapView()
        private var count = 1
        private var color = UIColor.orange
        private let mapMode = UISegmentedControl(items: ["Map", "Satellite", "Mix"])

        override init(frame : CGRect) {
            find.setTitle("Where am I?", for: .normal)
            location.isSelectable = false
            location.isEditable = false
            location.text = "Where am I?"
            location.textAlignment = .center
            lattitude.text = "Lat : --"
            lattitude.textAlignment = .left
            longitude.text = "Long : --"
            longitude.textAlignment = .right
            crtTime.text = "--"
            crtTime.textAlignment = .center
            map.isScrollEnabled = true
            map.isZoomEnabled = true
            CLmngr.distanceFilter = 1.0 // precision = 1m
            CLmngr.requestWhenInUseAuthorization()
            super.init(frame: frame)
            self.backgroundColor = UIColor.white
            find.addTarget(self, action: #selector(computePosition(sender:)), for: .touchDown)
            add.addTarget(self, action: #selector(addPin(sender:)), for: .touchDown)
            map.delegate = self
            CLmngr.delegate = self
            self.addSubview(find)
            self.addSubview(add)
            self.addSubview(location)
            self.addSubview(lattitude)
            self.addSubview(longitude)
            self.addSubview(crtTime)
            self.addSubview(map)
            mapMode.backgroundColor = UIColor(red: 1, green: 1, blue: 1, alpha: 0.5)
            mapMode.selectedSegmentIndex = 0
            mapMode.addTarget(self, action: #selector(changeMap(sender:)),
                              for: .valueChanged)
            self.addSubview(mapMode)
            self.drawInSize(UIScreen.main.bounds.size)
}
```

MyView (updated)


```
@objc func changeMap(sender: UISegmentedControl) {
    if sender.selectedSegmentIndex == 0 {
        map.mapType = .standard
    }
    if sender.selectedSegmentIndex == 1 {
        map.mapType = .satellite
    }
    if sender.selectedSegmentIndex == 2 {
        map.mapType = .hybrid
    }
}
```

```
required init?(coder aDecoder: NSCoder) { // Requested by Xcode
    fatalError("init(coder:) has not been implemented")
}

func drawInSize(_ size: CGSize) {
    var top = 20;
    if UIDevice.current.userInterfaceIdiom == .phone &&
        size.height >= 812 {
        top = 30
    } else if UIDevice.current.userInterfaceIdiom == .phone &&
        size.width > size.height {
        top = 0
    }
    find.frame = CGRect(x: Int(size.width / 2 - 50), y: top + 10, width: 100, height: 30)
    add.frame = CGRect(x: Int(size.width - 40), y: top + 10, width: 30, height: 30)
    crtTime.frame = CGRect(x: 10, y: top + 40, width: Int(size.width - 20), height: 20)
    lattitude.frame = CGRect(x: 10, y: top + 70, width: 120, height: 30)
    longitude.frame = CGRect(x: Int(size.width - 130), y: top + 70, width: 120, height: 30)
    location.frame = CGRect(x: 10, y: top + 100, width: Int(size.width - 20), height: 60)
    map.frame = CGRect(x: 0 , y: top + 160, width: Int(size.width), height: Int(size.height - 160))
    mapMode.frame = CGRect(x: 20 , y: top + 180,
                           width: Int(size.width - 40) , height: 30)
}

@objc func computePosition(sender : UIButton) {
    location.text = "I am searching..."
    CLmngr.startUpdatingLocation()
}
```

MyView (updated)


```
@objc func addPin(sender : UIButton) {
    let a = AnAnnotation(c: map.centerCoordinate,
                         t: String(format:"location %d", count),
                         st: "subtitle")
    count += 1
    map.addAnnotation(a)
}

func nextColor (c : UIColor) -> UIColor {
    switch c {
        case .orange : return .red
        case .red : return .blue
        case .blue : return .green
        case .green : return .black
        case .black : return .purple
        default : return .orange
    }
}

// CLLocationManagerDelegate protocol

func locationManager(_ manager: CLLocationManager,
                     didUpdateLocations locations: [CLLocation]) {
    lattitude.text = String(format: "Lat : %2.4f",
                            (manager.location?.coordinate.latitude)!)
    longitude.text = String(format: "Long : %2.4f",
                            (manager.location?.coordinate.longitude)!)
    let myDate = DateFormatter()
    myDate.dateFormat = "dd-MM-yyyy, HH'h'mm"
    crtTime.text = myDate.string(from: (manager.location?.timestamp)!)
    location.text = manager.location?.description
    CLmngr.stopUpdatingLocation() // Only one measure
    // Map update
    let span = MKCoordinateSpan(latitudeDelta: 0.035, longitudeDelta: 0.035)
    let region = MKCoordinateRegion(center: (manager.location?.coordinate)!,
                                    span: span)
    map.setRegion(region, animated: true)
    map.showsUserLocation = true // Add a pin on the current location
}

func locationManager(_ manager: CLLocationManager,
                     didFailWithError error: Error) {
    lattitude.text = "Lat : --"
    longitude.text = "Long : --"
    crtTime.text = "--- error ---"
    location.text = error.localizedDescription
}

func mapView(_ mapView: MKMapView,
             viewFor annotation: MKAnnotation) -> MKAnnotationView? {
    let epingle = MKPinAnnotationView(annotation: annotation,
                                      reuseIdentifier: "ppm")
    epingle.pinTintColor = color
    color = nextColor(c:color)
    epingle.canShowCallout = true
    epingle.rightCalloutAccessoryView = UIButton(type : .detailDisclosure)
    return epingle
}
```

MyView (updated)


```
    func mapView(_ mapView: MKMapView,
                 annotationView view: MKAnnotationView,
                 calloutAccessoryControlTapped control: UIControl) {
        location.text = "coordinate of «" + (view.annotation?.title!)! + "»"
        let c = view.annotation?.coordinate
        lattitude.text = String (format:"Lat : %2.4f", (c?.latitude)!)
        longitude.text = String (format:"Long : %2.4f", (c?.longitude)!)
    }

    func mapView(_ mapView: MKMapView, didSelect view: MKAnnotationView) {
        location.text = "You selected «" + (view.annotation?.title!)! + "»"
    }
}
```

As a conclusion…

Is'n't it easy?

- Especially compared to the management of pins in maps
- Your apps are ready to explore the world…