

IOS14

Youssef  
BALOUKI

GI 2020

# Introduction

## Qu'est-ce que l'iPhone ?

**iPhone** est une gamme de smartphones commercialisée par Apple depuis le 29 juin 2007.

- ✓ Processeur
- ✓ Système d'exploitation : **iOS**
- ✓ Support réseau : EDGE/CDMA/3G/LTE/4G, Wifi, Bluetooth
- ✓ Ecran tactile capacitif multipoint
- ✓ Appareil photo qui fonctionne également comme une caméra,
- ✓ Système de géolocalisation intégré
- ✓ iPod intégré qui permet bite d'écouter et de télécharger de la musique
- ✓ Gyroscope
- ✓ Accéléromètre
- ✓ Capteur de luminosité ambiante
- ✓ Etc....



# Introduction

## iOS

- ❑ Le système d'exploitation développé par Apple pour ses appareils mobiles : l'iPhone, de l'iPad, ainsi que de l'iPod Touch.
- ❑ Il est dérivé de OS X
- ❑ Anciennement iPhone OS, Ce nom a été changé le 7 juin 2010 pour iOS

# Introduction

## Xcode

- **Les modules**

- ✓ StoryBoard
- ✓ Simulateur/Debugger
- ✓ Instrument Profiler
- ✓ Analyse statistique
- ✓ Gestionnaire de versions intégrés
  - Git, Subversion
- ✓ Accès directe à la documentation

# Construction des interfaces

## Deux approches

### ❑ Une approche basée sur storyboard

- ✓ Par dessin comme
- ✓ On ne maîtrise pas totalement la construction de l'interface

### ❑ Une approche par pure programmation

- ✓ On fait tout à la main
- ✓ Maîtrise complet du design de l'interface
- ✓ Mais plus compliqué (réservé aux experts)

# Les bases de Storyboard

## ■ Interface graphique

- ✓ Dessiner ses écrans
- ✓ Positionnement par contraintes
- ✓ Scénariser l'application
- ✓ Lier les éléments de l'interface au code métier

## ■ Génération de code “AutoLayout”

- ✓ Introduit avec iOS 6 (2012)
- ✓ Positionnement relatif des éléments de l'interface
  - Relations entre les éléments
  - Contraintes par rapport à l'écran

# Modèles de iPhone

- iPhone
  - iPhone 3G
  - iPhone 3GS
  - iPhone 4
  - iPhone 4s
  - iPhone 5
  - iPhone 5c
  - iPhone 5s
  - iPhone 6
  - iPhone 6Plus
  - iPhone 6s
  - iPhone 6s Plus
  - iPhone SE
- iPhone 7 et 7 Plus  
iPhone 8 et 8 Plus  
iPhone X  
iPhone XS et XS Max  
iPhone XR  
iPhone 11, 11 Pro et  
11 Pro Max  
iPhone SE - 2de gen



# Version iOS

- Juin 2007 v 1.x
- Juillet 2008 v 2.x
- Juin 2009 v 3.x
- Juin 2010 v 4.x
- Oct 2011 v 5.x
- Sept 2012 v6.x
- Juin 2013 v7.x
- Juin 2014 v8.x
- juin 2015 v9.x
- Sept 2016 v10.x
- Juin 2017 v11.x
- Juin 2018 v12.x
- juin 2019 v13.x



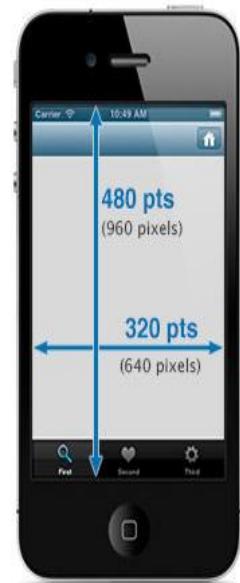
iOS 14

# Les tailles des écrans

iPhone 4, 4S

3.5" Screen

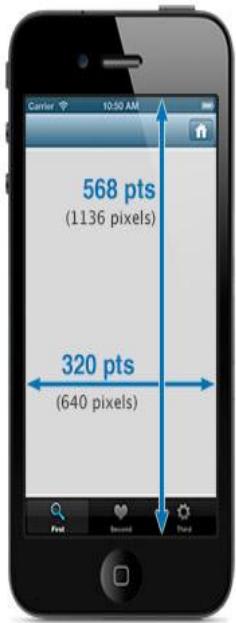
@2x images



iPhone 5, SE

4" Screen

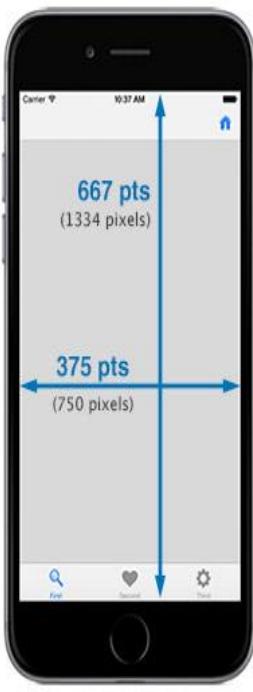
@2x images



iPhone 7, 8

4.7" Screen

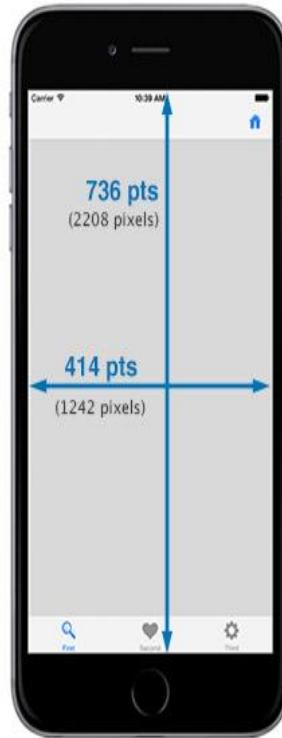
@2x images



iPhone 7 Plus, 8 Plus

5.5" Screen

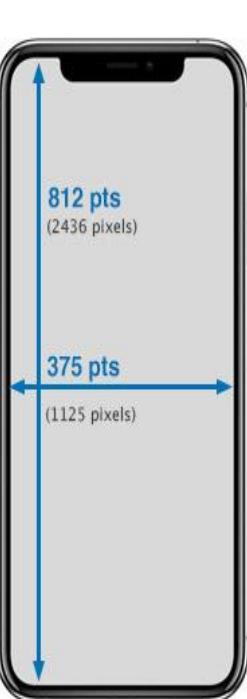
@3x images



iPhone X, Xs

5.8" Screen

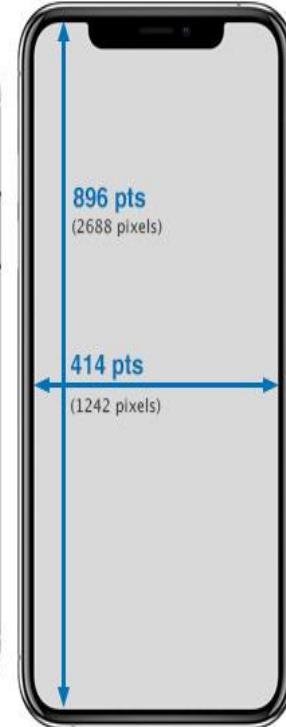
@3x images



iPhone Xs Max

6.5" Screen

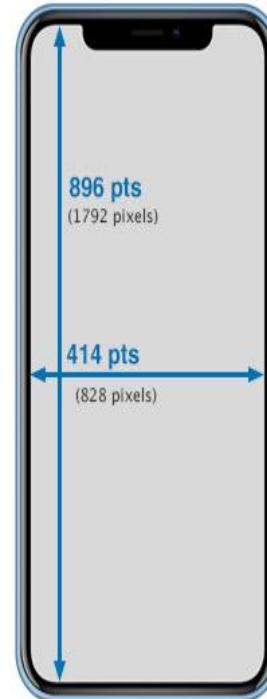
@3x images



iPhone XR

6.1" Screen

@2x images



# Rejoindre le programme développeur Apple

## • Inscription gratuite

- Vous donne votre **identifiant développeur Apple** (AppleID).
- Vous donne accès aux outils logiciels de base (comme **Xcode** et **iOS SDK**) dont vous avez besoin pour développer des applications pour iOS.
- Vous permet de tester vos applications iOS en utilisant des **simulateurs** dans Xcode et **devices**
- Vous fournit la documentation Apple relative au développement iOS.
- Vous permet de signaler les bugs et de participer aux événements Apple

# Rejoindre le programme développeur Apple

- **Inscription payée (nécessite un paiement annuel de 99 \$ à Apple)**
  - Vous permet d'accéder aux outils développeur, aux logiciels pré-release et au support technique.
  - Vous permet de placer vos applications iOS achevées dans **'App Store'**.
  - Vous permet d'accéder au portail par le biais duquel vous créez le **profil d'approvisionnement de développement**, qui est nécessaire à l'exécution de vos applications sur des périphériques iOS de développement.

# Structure de l'iOS

## Cocoa Touch

Multi-touch, Core Motion, Camera, Map Kit, Controls, ...

## Media Services

Quartz, Core Animation, Open GL ES, ...

## Core Services

Collections, Address Book, Threading, Core Location...

## Core OS

Kernel, Filesystem, Security, Sockets, Certificates, ..

## **■ Interactions**

**UIKit, Événements tactiles**

## **■ Multimédia**

**Traitement images, Audio, Vidéo, Animations, 2D, 3D**

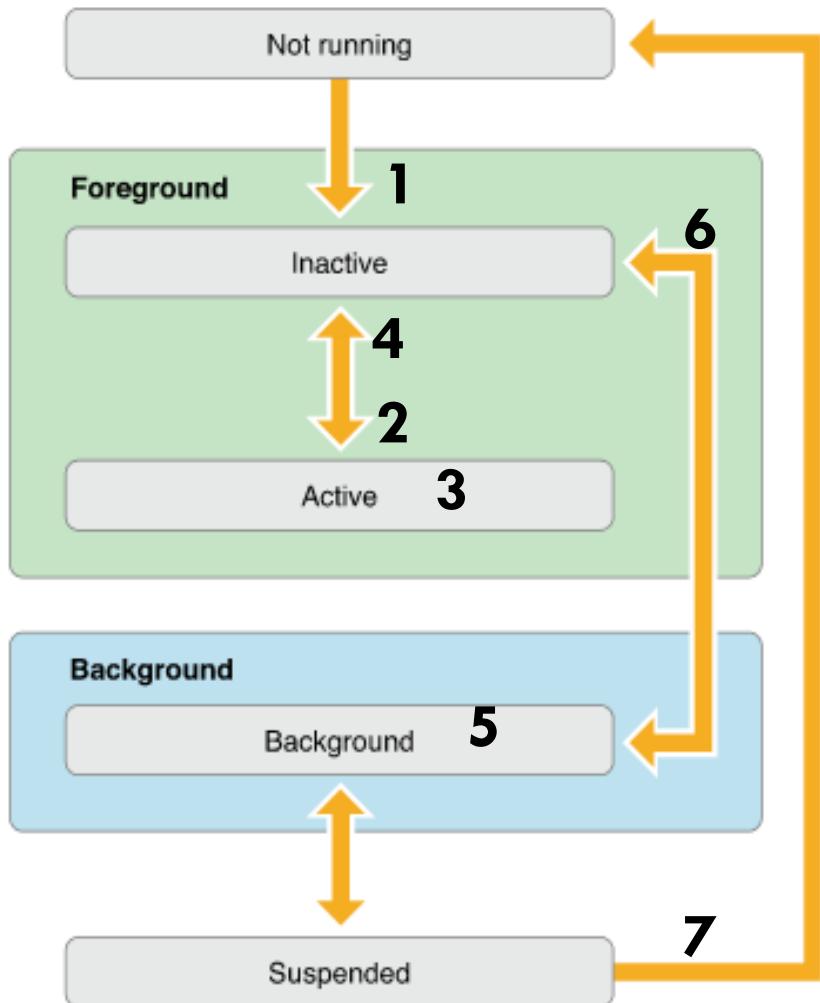
## **■ Expose les fonctionnalités du système**

**Réseau, Fichiers, SQLite, Threads, GPS**

## **■ Noyau Unix**

**Fonctionnalités primaire du système d'exploitation Sécurité, Alimentation, Sockets**

# Cycle de vie de l'application iOS



1. `application:willFinishLaunchingWithOptions:`
2. `application:didFinishLaunchingWithOptions:`
3. `applicationDidBecomeActive:`
4. `applicationWillResignActive:`
5. `applicationDidEnterBackground:`
6. `applicationWillEnterForeground:`
7. `applicationWillTerminate: si != suspended`

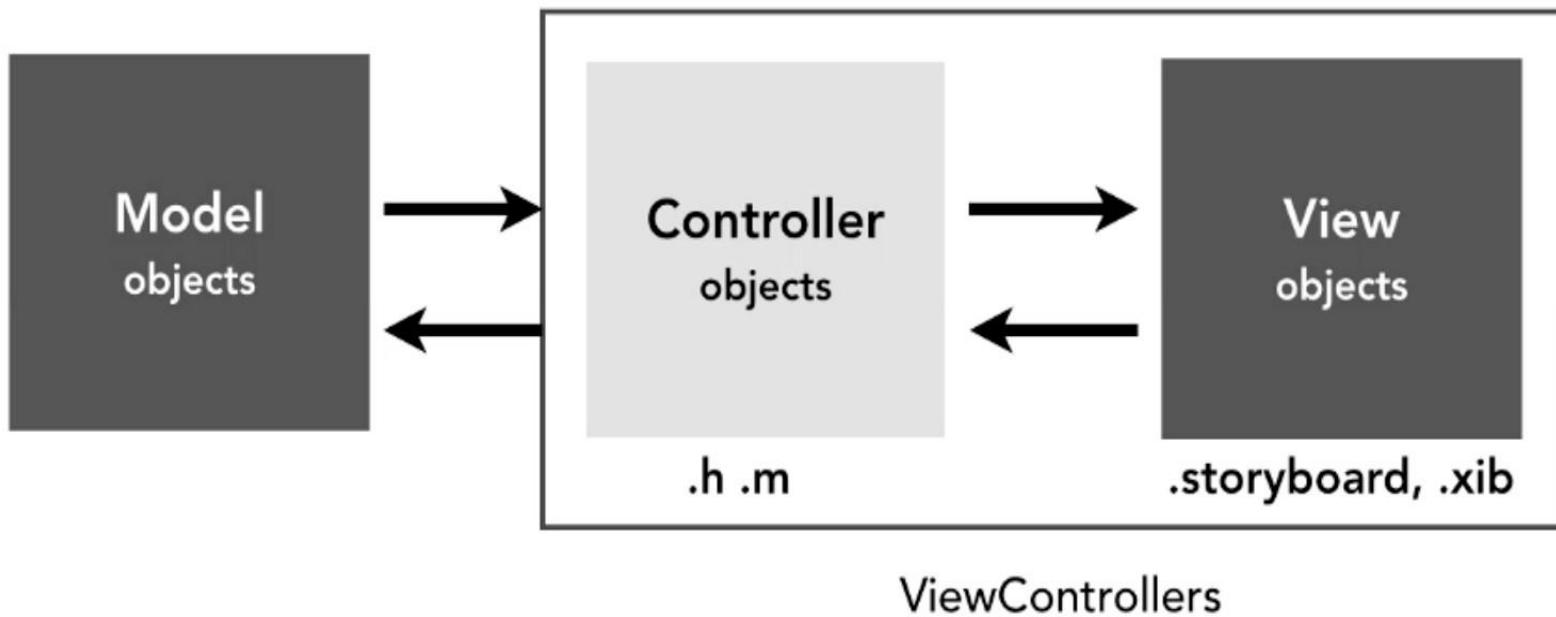
# Interface Graphique

Dans l'iOS en utilise le **MVC**.

**Modèle , view, contrôleur**

- **Modèle:**  
Quelques sont vos données?
- **Vue: UIView (Storyboard)**  
Où sont présentées les données?
- **Contrôleur UIViewController**  
Comment sont présentées les données?

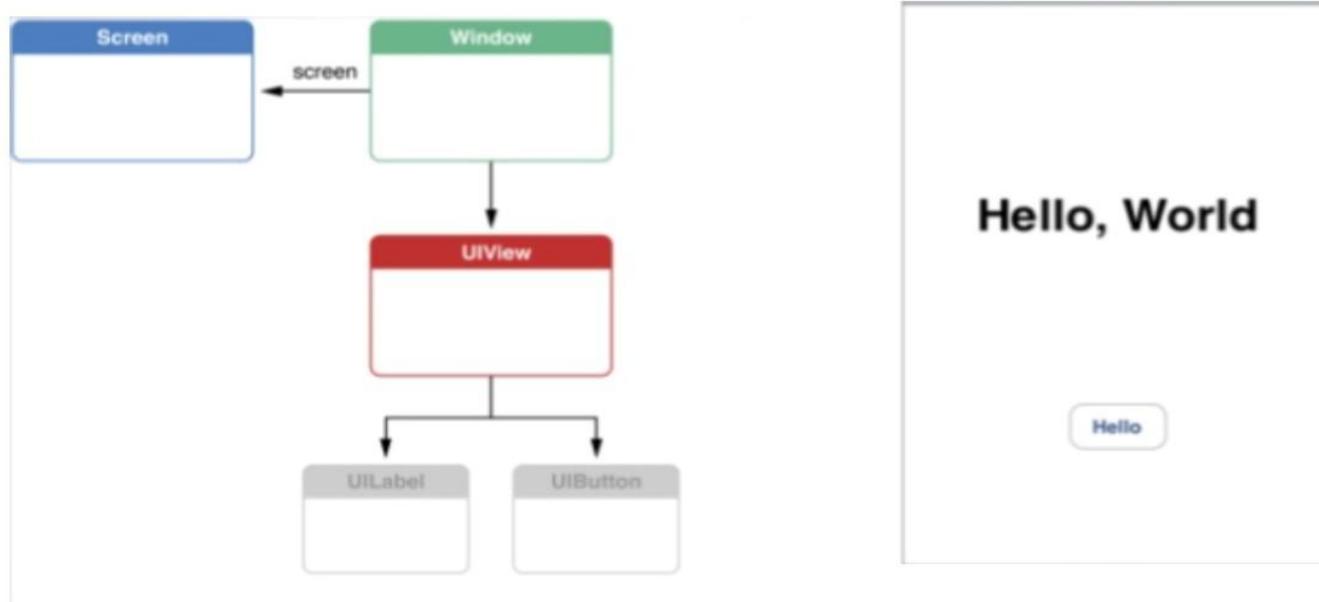
# Interface Graphique



# Interface Graphique

## UIViews

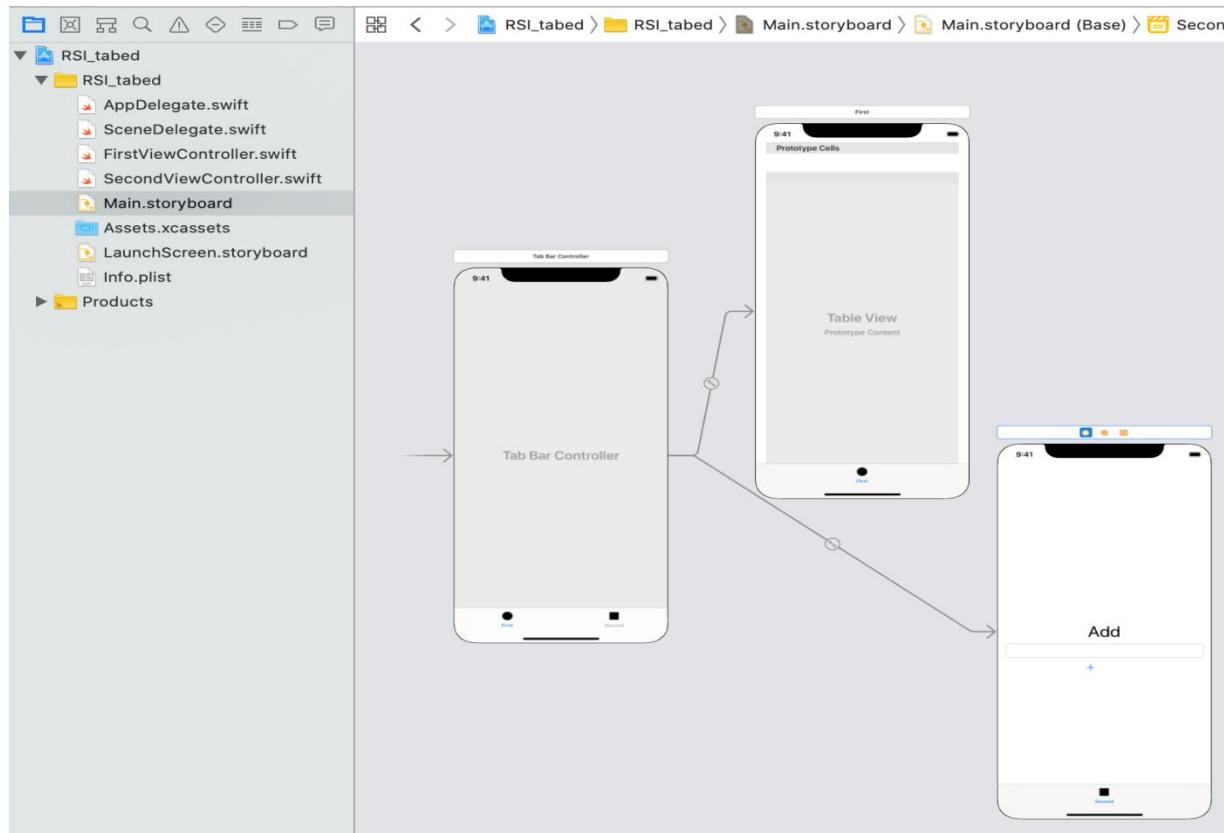
- Une **UIView** : est une surface rectangulaire dans laquelle on peut dessiner et gerer les évènements.
- Hierarchie : une vue a une superview et des sous- views.



# Interface Graphique

## Storyboard

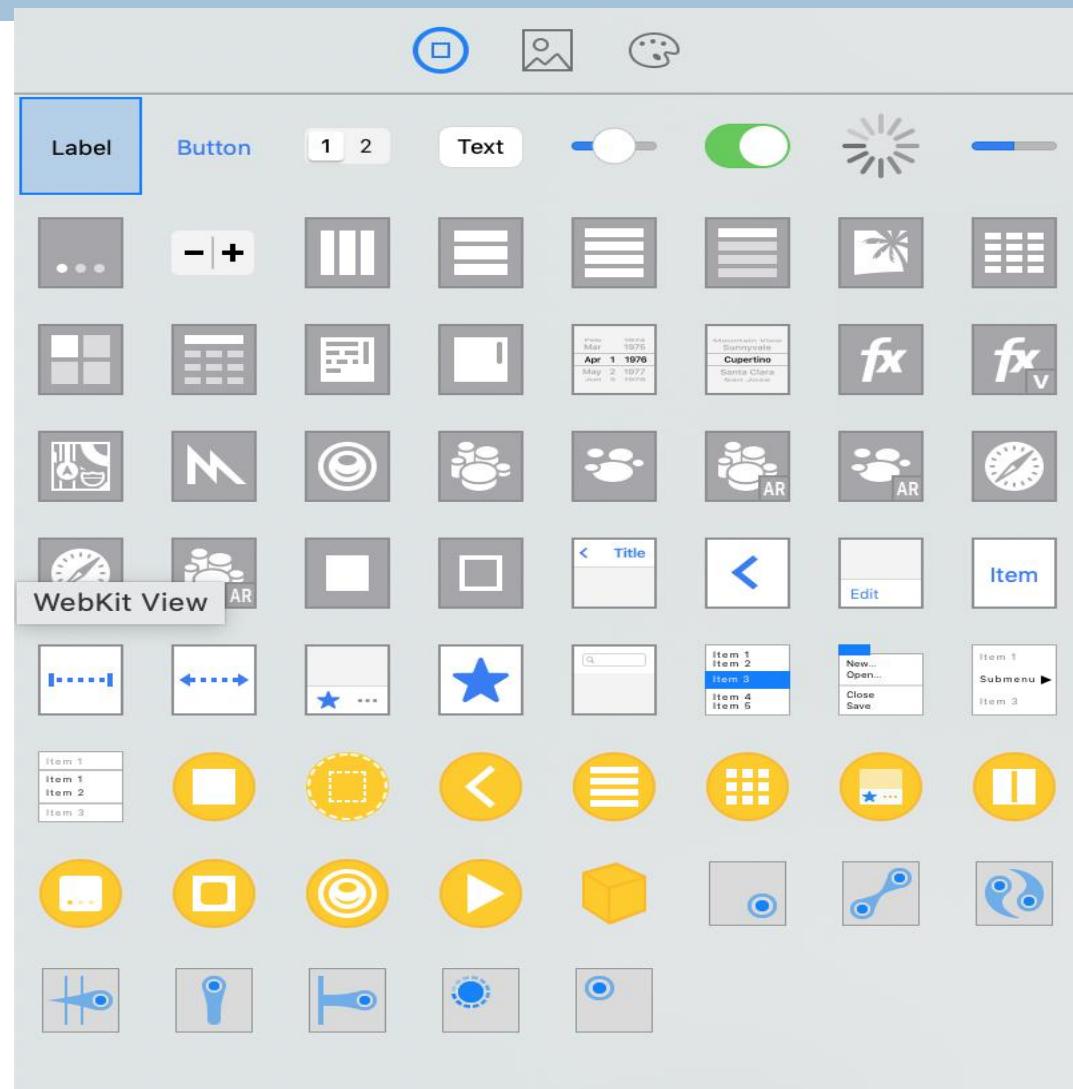
- **Storyboard** un fichier qui contient tous les écrans de l'application et les liens établis entre chaque écran.



# Interface Graphique

## Widgets

- UIButton
- UITableView
- UITextView
- UIImageView
- UIWebView
- UIScrollView
- UILabel
- UITextField
- ....



# Interface Graphique

- **Les outlets:** propriétés du code sources liées à un élément de l'interface

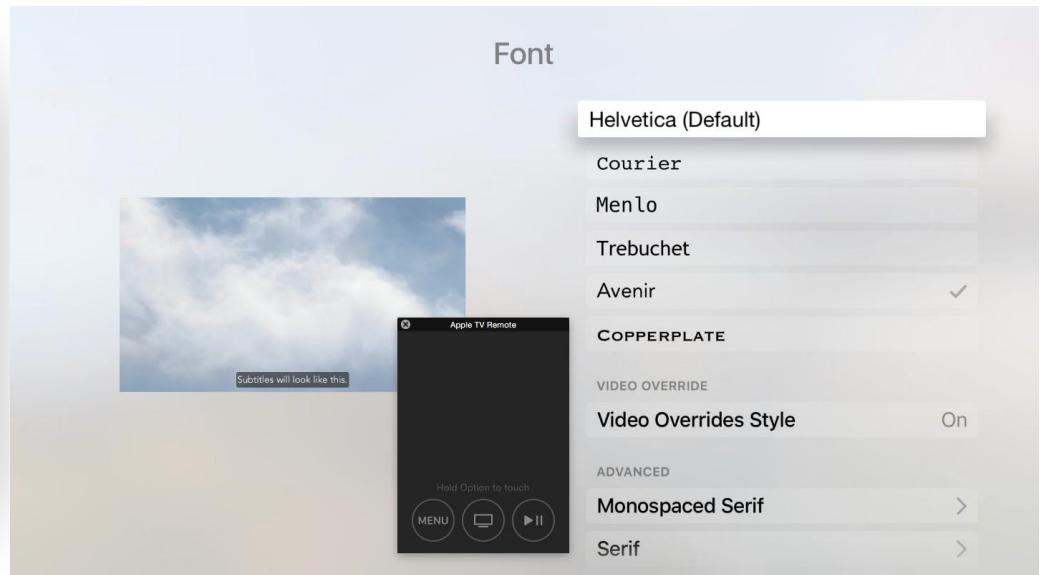
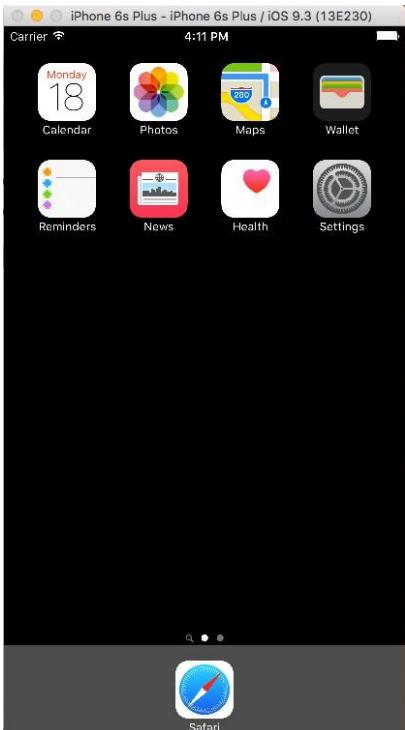
**Exemple:** button, label ....

- **Actions:** Les messages quand un événement se produit

**Exemple:** “une touche sur le button ”

# LE SIMULATEUR IOS

- Le simulateur est une application Mac permet de simuler le fonctionnement d'un iPhone, d'un iPod Touch,iPad...



# LE SIMULATEUR IOS

Si vous voulez utiliser le simulateur iOS, vous pouvez à tout moment changer de device en utilisant la liste déroulante “**Scheme**”



# LE SIMULATEUR IOS

## Simuler les gestuelles de l'utilisateur

vous pouvez utiliser des commandes dans le menu **Matériel** du simulateur iOS, ou les raccourcis clavier équivalents, pour simuler les actions dans le simulateur iOS

Action	Commande dans le menu Matériel	Raccourci
Rotation d'un quart de tour à gauche	Rotation à gauche	<b>Commande</b> + <b>Gauche</b>
Rotation d'un quart de tour à droite	Rotation à droite	<b>Commande</b> + <b>Droite</b>
Verrouillage	Verrouiller	<b>Commande</b> + <b>L</b>
Secousse	Secousse	<b>Ctrl</b> + <b>Commande</b> + <b>Z</b>
Ecran d'accueil	Ecran d'accueil	<b>Maj</b> + <b>Commande</b> + <b>H</b>

# Mode Visuel

1. Choisir une vue à partir du bibliothèque
2. L'ajouter dans le storyBoard
3. La mise en forme de la nouvelle vue dans le
4. Story board en respectant les **# device et # orientations**

# Label, TextField & Button

✓ **UITextField, UIButton et UILabel héritent de UIView.**

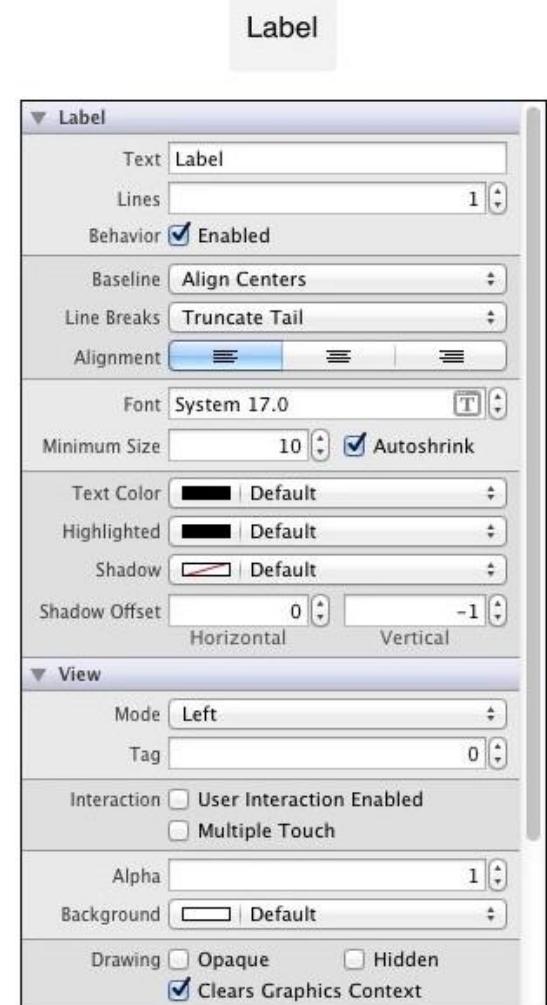
Icône	Nom du contrôle	Fonction
Label	Label	Texte non modifiable par l'utilisateur
Text	Text Field	Zone de texte modifiable par l'utilisateur
Round Rect Button	Round Rect Button	Bouton de commande touch
Text View	Text View	Zone de texte multiligne éditable

# Label, TextField & Button

## Les contrôles

### **UILabel**

- Les contrôles **Label** sont utilisés pour afficher des textes sur une ou plusieurs lignes. Leur contenu ne peut pas être modifié par l'utilisateur.
- Les caractéristiques d'un Label peuvent être définies dans Interface Builder ou bien avec le code.



# Label, TextField & Button

**Exemple :**

```
monLabel.text = "Un court texte affiché dans le  
contrôle Label sur deux lignes"
```

```
monLabel.numberOfLines = 2
```

```
monLabel.font = UIFont( name:"Courier", size : 14)  
monLabel.textAlignment = .center
```

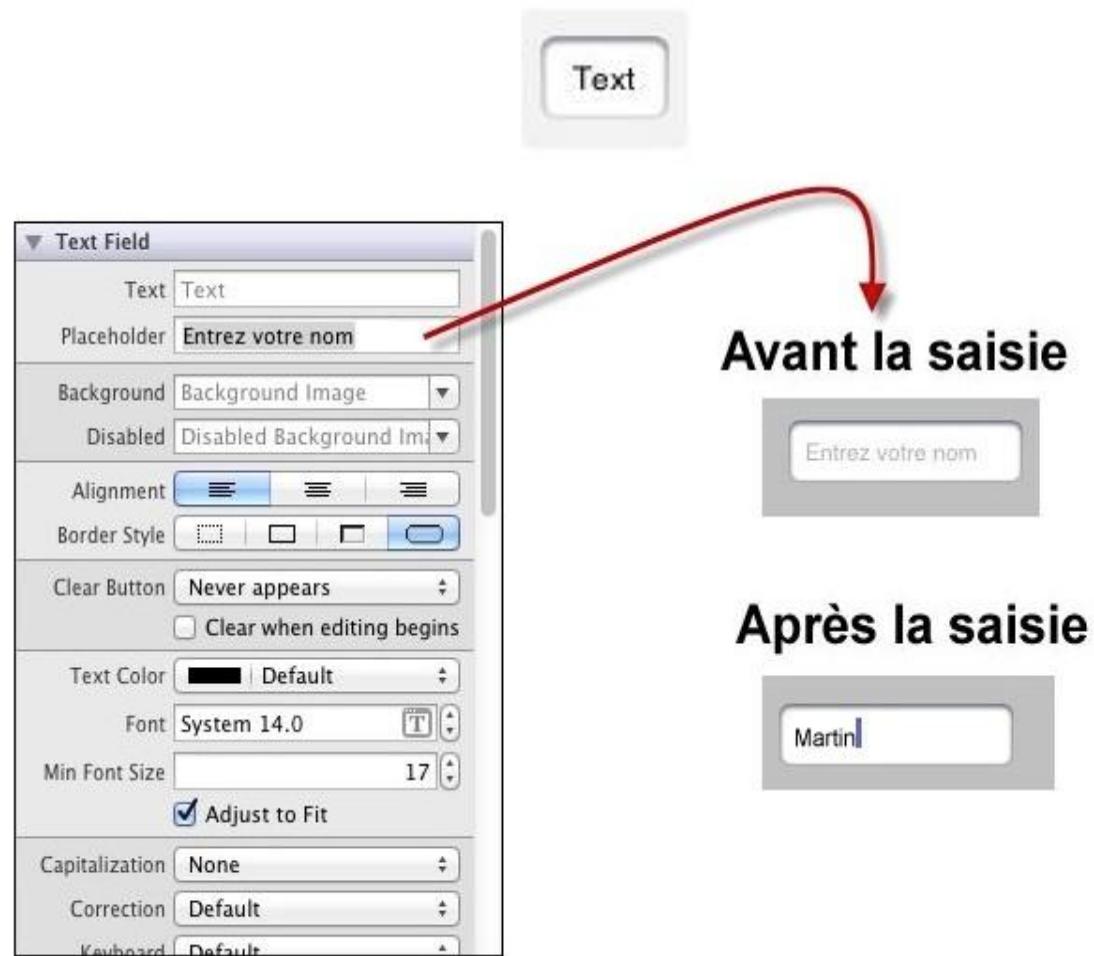
```
monLabel.textColor = UIColor (red: 1.0/255.0,  
green: 0.0/255.0 blue: 0.0/255.0 alpha: 1.0)
```

# Label, TextField & Button

## Les contrôles

### UITextField

- Les Text Field sont des zones de texte monoligne.
- Ils sont utilisés pour saisir des données textuelles courtes, comme un nom ou un mot de passe
- Tout comme pour les Labels, vous pouvez choisir l'alignement, la police et la couleur du texte, dans Interface Builder ou avec du code



# Label, TextField & Button

**Exemple :**

`monTextField.text = Un court texte affiché dans le  
contrôle Label sur deux lignes"`

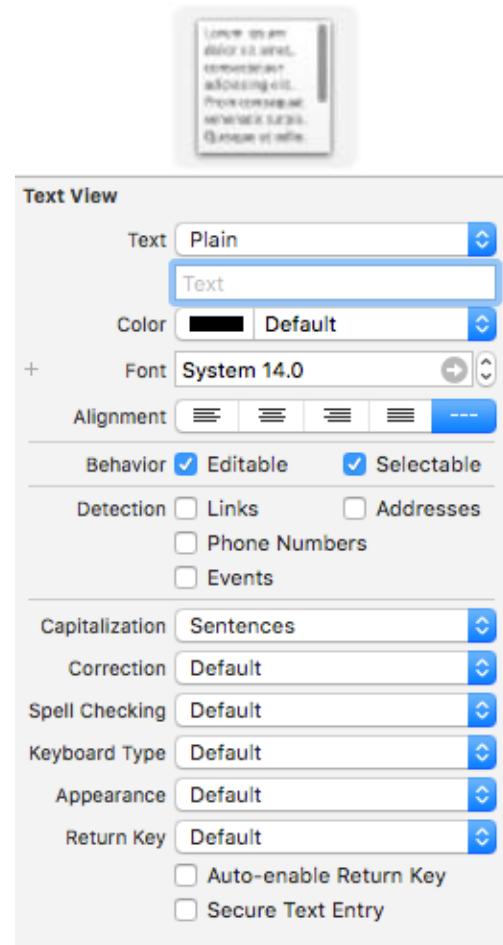
`monTextField.placeholder = "Entrez votre nom"`

# Label, TextField & Button

## Les contrôles

### UITextView

- Les Text View sont des zones de texte multiligne.
- Ce contrôle peut être affiche en lecture seule ou en lecture/ecriture.



# UILabel, UITextField & UIButton

The image shows a development environment with two main windows. On the right is the Xcode code editor, displaying Swift code for a View Controller. On the left is an iPhone 6 simulator window showing the application's interface.

**Xcode Code Editor:**

```
import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var libelle: UILabel!

    @IBOutlet weak var zone_texte: UITextField!

    @IBAction func action_afficher(_ sender: UIButton) {
        libelle.text = zone_texte.text
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

**iPhone Simulator:**

The simulator displays a simple UI with a label "salut tout le monde" and a text field containing the same text. A button labeled "Afficher" is present. The keyboard is visible at the bottom.

# Application : UIButton & UITextField

The screenshot shows the Xcode interface with a Swift file named ViewController.swift open. The code implements a simple number guessing game where the user inputs a number and receives feedback ("Plus grand", "Plus petit", or "Bravo!"). The Xcode interface includes a preview window showing the app running on an iPhone 6 simulator.

```
// Copyright © 2017 youssef balouki. All rights reserved.

import UIKit

class ViewController: UIViewController, UITextViewDelegate {

    @IBOutlet weak var txtNbr: UITextField!
    var nbrGenerer : Int = 0
    var msg : String = ""

    @IBOutlet weak var txt_resultat: UITextField!
    @IBAction func btnVerfication(_ sender: UIButton) {
        let nbr = Int(txtNbr.text!)

        if nbr! > nbrGenerer{ txt_resultat.text = "Plus grand" }
        else if nbr! < nbrGenerer{ txt_resultat.text = "Plus petit" }
        else{ txt_resultat.text = "Bravo!" }

    }

    override func viewDidLoad() {
        super.viewDidLoad()

        nbrGenerer = Int(arc4random_uniform(10))

    }

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

    override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
        view.endEditing(true)
    }
}
```

iPhone 6 – iOS 10.1 (14B72)  
Carrier 10:28 PM  
Dvinette !!!  
2  
Verifier  
Plus petit

# Alerte & Actions

```
// Created by youssef balouki on 14/04/17.
// Copyright © 2017 youssef balouki. All rights reserved.
import UIKit

class ViewController: UIViewController {

    @IBAction func action_button(_ sender: UIButton) {
        let alert = UIAlertController(title : "Titre d'Alert"
            ,message : "Message d'Alert", preferredStyle: .alert)

        let OkAction = UIAlertAction(title : "Fin"
            ,style: .default){
            (action:UIAlertAction!) in
        }

        alert.addAction(OkAction)

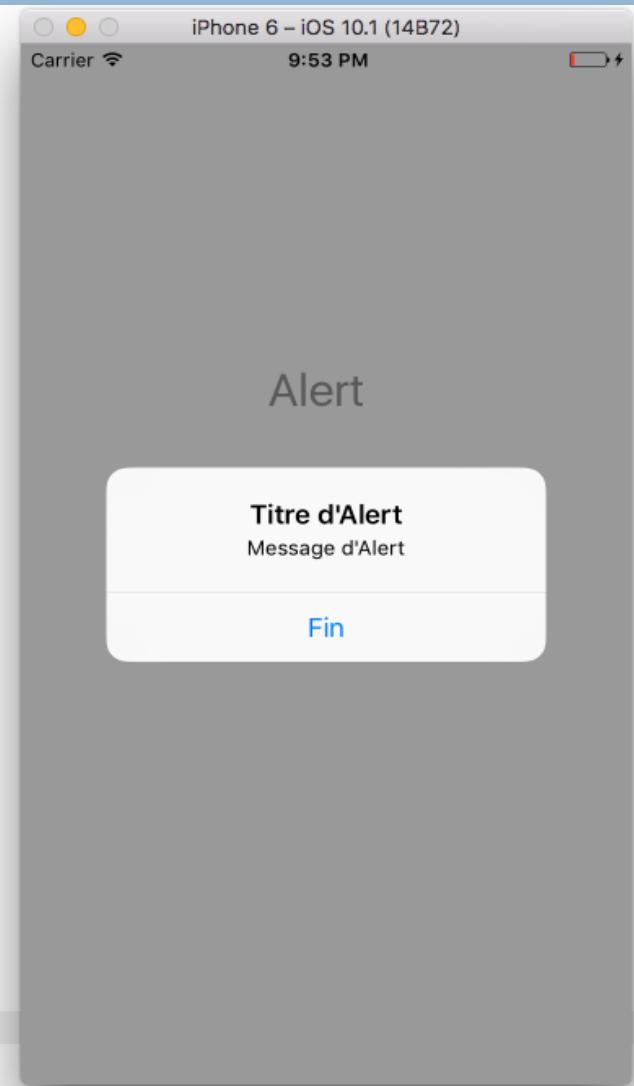
        self.present(alert
            , animated: true
            ,completion: nil)

    }

    override func viewDidLoad() {
        super.viewDidLoad()

    }

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



# Application : Devinette avec alertes et actions

Deviner > iPhone 7 Plus      Running Deviner on iPhone 7 Plus

ViewController.swift

```
import UIKit
class ViewController: UIViewController, UITextFieldDelegate {

    let randnumber = arc4random_uniform(10)
    var compteur = 0
    @IBOutlet weak var txtnb: UITextField!
    @IBOutlet weak var lblcompteur: UILabel!
    @IBOutlet weak var lblresultat: UILabel!

    @IBAction func act(_ sender: UIButton) {
        compteur += 1
        if UInt32(txtnb.text!)! < randnumber {
            lblresultat.text = "Plus grand"
            popUp(titre: "Alert", message: "Plus grand")
        }else if UInt32(txtnb.text!)!>randnumber{
            lblresultat.text = "Plus petit"
            popUp(titre: "Alert", message: "Plus petit")
        }else {
            lblresultat.text = "Bravo"
            popUp(titre: "Alert", message: "Bravo!!!")
        }
        lblcompteur.text = String(compteur)
    }
    func popUp(titre : String, message : String) {

        let alrt = UIAlertController(title: titre, message: message, preferredStyle: .alert)
        let act = UIAlertAction(title: "Cancel", style: .cancel, handler: nil)
        alrt.addAction(act)
        self.present(alrt, animated: true, completion: nil)
    }
    override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
        view.endEditing(true)
    }
    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        txtnb.resignFirstResponder()
        return true
    }
}
```

iPhone 7 Plus – iOS 10.1 (14B72)

Carrier 1:05 AM

Devinette

2

Lancer

Message : Plus petit

Tentatives : 1

«2»

a z e r t y u i o p  
q s d f g h j k l m  
w x c v b n ,  
123 , espace retour

# Slider, Stiper & Segmented

Icône	Nom du contrôle	Fonction
	Slider	Curseur pour faciliter la saisie d'une valeur
	Segmented Control	Onglets permettant d'afficher différentes vues/contrôles
	Image View	Conteneur permettant d'afficher une image ou une animation
	Switch	Bouton de type ON/OFF

# UISlider, UISlider & UISegmented

## Le slider : (de classe UISlider)

la vue qui vous permet de choisir une valeur entre **0** et **99**. Bien sûr, vous pouvez choisir les valeurs que vous voulez.

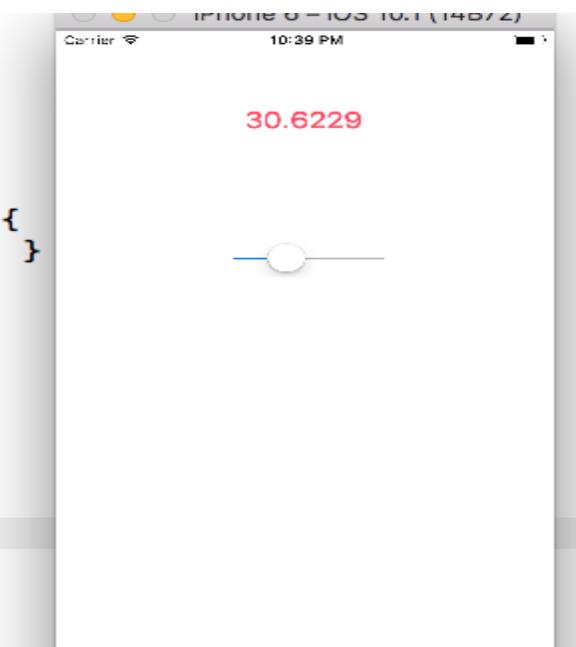
```
import UIKit

class ViewController: UIViewController {
    @IBOutlet weak var libelle: UILabel!

    @IBAction func action(_ sender: UISlider) {
        libelle.text = String(sender.value)
    }

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

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



# UISlider, UIStepper & UISegmented

## Le steper : (de classe UIStepper)

Un "+" et un "-" qui incrémente ou décrémente une valeur. Valeurs comprises entre 0 et 99 également avec un pas de 1.

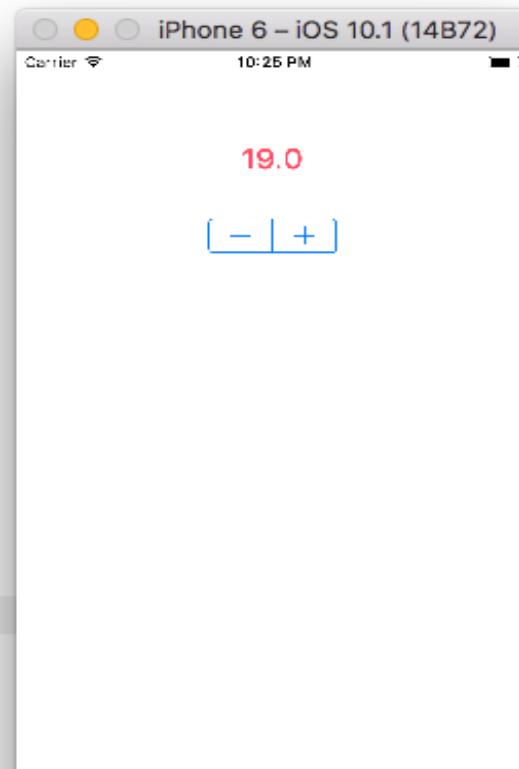
```
import UIKit

class ViewController: UIViewController {
    @IBOutlet weak var libelle: UILabel!

    @IBAction func stiper(_ sender: UIStepper) {
        libelle.text = String(sender.value)
    }
    override func viewDidLoad() {
        super.viewDidLoad()

    }

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



# Application : UISlider & UISegment

The screenshot shows the Xcode interface with the following details:

- Project Structure:** mars\_slider > mars\_slider > ViewController.swift
- Code (ViewController.swift):**

```
// Copyright © 2017 youssef balouki. All rights reserved.
import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var sgm: UISegmentedControl!
    @IBOutlet weak var sgmd: UISegmentedControl!
    @IBOutlet weak var lbl: UILabel!
    @IBOutlet weak var lblh: UILabel!

    @IBAction func sld(_ sender: UISlider) {
        lbl.text = String(Int(sender.value))
        lblh.text = String(Int(sender.value), radix: 16)

        sgm.selectedSegmentIndex = Int(sender.value) % 10
        sgmd.selectedSegmentIndex = Int(sender.value) / 10
    }

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

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

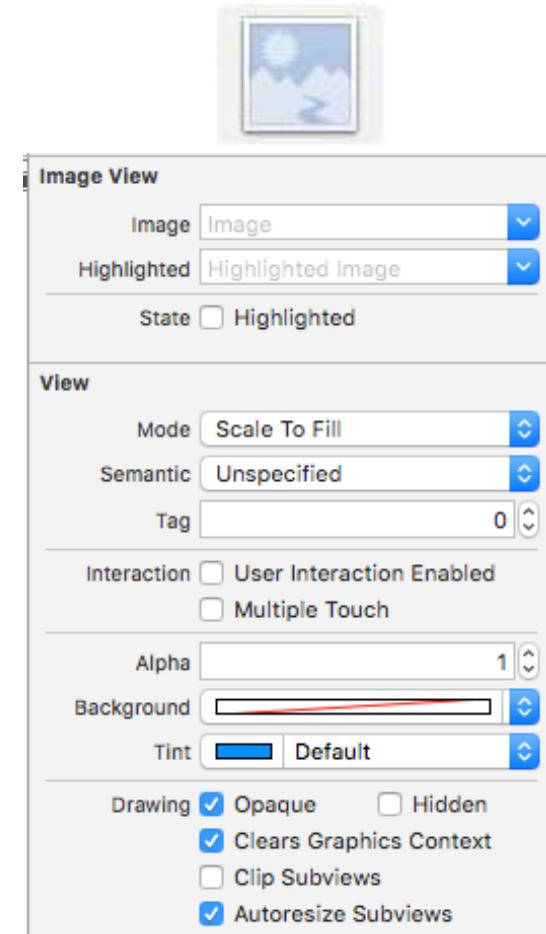
- Simulator View:** iPhone 6 – iOS 10.1 (14B72) at 1:19 AM. It displays:
  - A UISlider with its value set to 17.
  - Two UISegmentedControls below it. The first (sgm) has segments labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, with segment 7 selected. The second (sgmd) has segments labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, with segment 11 selected.
  - Text labels "Decimal : 17" and "Hexadecimal : 11" indicating the current values.

# FENÊTRES, VUES ET CONTRÔLES

## Les contrôles

### **UIImageView**

- Le contrôle **Image View** permet d'afficher une image.
- Les images sont chargées via des objets **UIImage**.



# FENÊTRES, VUES ET CONTRÔLES

Exemple :

Image dans Assets.xcassets

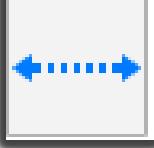
```
@IBOutlet image : UIImageView!  
image.image = UIImage(named : "monImage")
```

Image dans bundle

```
let path = Bundle.main.path(forResource : "imageName", ofType : "jpg")  
image.image = UIImage(contentsOfFile : path)
```

\*(Allow Arbitrary Loads key to YES under App Transport Security Settings dictionary in your .plist file)

# Navigation Bar & Toolbar

Icône	Nom du contrôle	Fonction
	Navigation Bar	Barre de navigation, affichée juste en dessous de la barre d'état
	Navigation Item	Elément affiché dans un contrôle Navigation Bar
	Toolbar	Barre d'outils contenant un ou plusieurs boutons
	Flexible Space Bar Button Item	Espace qui s'ajuste automatiquement en fonction de la place disponible dans un contrôle Toolbar

# Navigation Bar - Tool Bar

```
import UIKit

class ViewController: UIViewController {

    var time = Timer()
    var compte = 0

    @IBOutlet weak var img: UIImageView!
    @IBOutlet weak var lbl: UILabel!

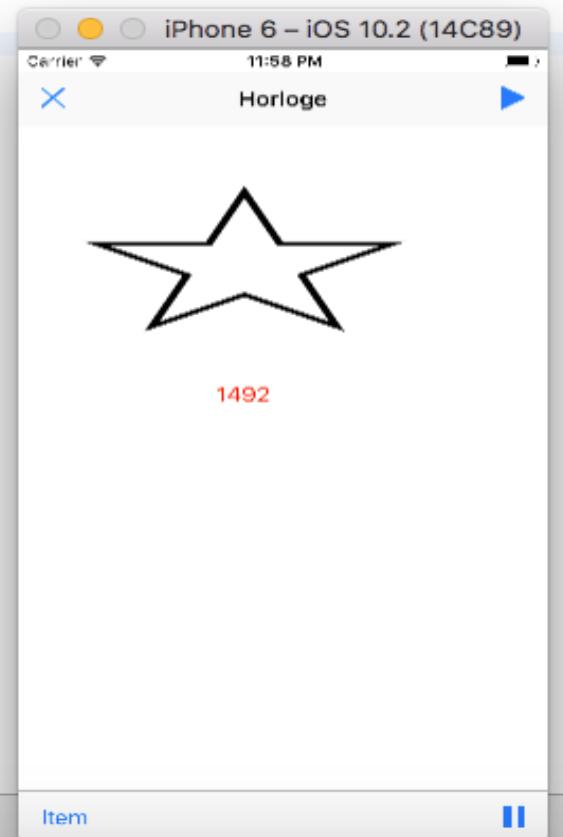
    @IBAction func play(_ sender: UIBarButtonItem) {
        time = Timer.scheduledTimer(timeInterval: 1, target: self, selector: #selector(affiche), userInfo: nil, repeats: true)
    }

    @IBAction func stop(_ sender: UIBarButtonItem) {
        time.invalidate()
    }

    func affiche() {
        compte += 1
        lbl.text = String(compte)
        if (compte % 2) == 0 {
            img.image = UIImage(named: "star")
        } else {
            img.image = UIImage(named: "starNoir")
        }
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



# LES INFORMATIONS TABULAIRES

Les applications iOS ont souvent recours à des vues tabulaires pour présenter des listes de données. Ces listes permettent de :

- présenter un ensemble d'options sélectionnables par l'utilisateur ;
- naviguer dans un ensemble de données structurées hiérarchiquement ;
- présenter une liste d'éléments triés selon un ou plusieurs critères ;
- définir plusieurs niveaux de détail pour faciliter l'affichage des informations sur des devices de petite taille.

# LES INFORMATIONS TABULAIRES

- **Table View** :affiche des listes d'informations sur une colonne ;
- **Picker View** :affiche des informations sur une ou plusieurs colonnes, sous la forme d'une roulette 3D
- **Date Picker** : similaire au deuxième, il est spécialisé dans 'affichage des dates et d'heures.

# Sélection parmi des choix multiples

## UIPickerView

**Objectif** : permettre aux utilisateurs de votre application de sélectionner à partir d'une liste de valeurs.

1. Ouvrir **Storyboard**, glisser un **UIPickerView** & **UITextField**
  2. Créer un **IBOutlet** pour UIPickerView & UITextField
  3. Lier UIPickerView & UITextField puis lier le UIPickerView à une **dataSource** & **delegate**
1. Importer les fonctions datasource et delegate requises
  2. Définir les fonctions datasource et tester
  3. Créer un tableau de données puis afficher les données en utilisant ce tableau.

# sélection parmi des choix multiples

## UIPickerView

```
s ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
@IBOutlet var txt : UITextField!

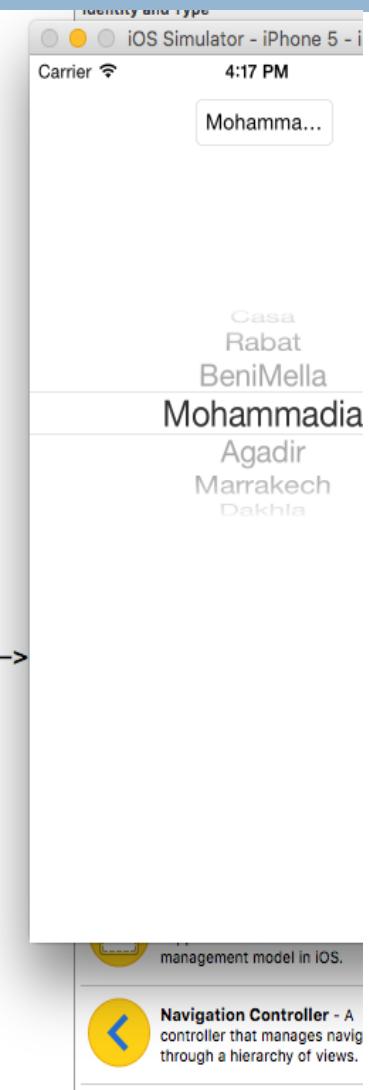
var tableau = ["Casa","Rabat","BeniMella","Mohammadia","Agadir","Marrakech","Dakhla"]
@IBOutlet var picker : UIPickerView!

// returns the number of 'columns' to display.
func numberOfComponentsInPickerView(pickerView: UIPickerView) -> Int{
    return 1
}

// returns the # of rows in each component..
func pickerView(pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int{
    return tableau.count
}

func pickerView(pickerView: UIPickerView, titleForRow row: Int, forComponent component: Int) ->
String!
{
    return tableau[row]
}

func pickerView(pickerView: UIPickerView, didSelectRow row: Int, inComponent component: Int){
    txt.text = tableau[row]
}
override func viewDidLoad() {
    super.viewDidLoad()
}
```



Youssef  
BALOUKI

# Sélection parmi des choix multiples

## UIPickerView

```
import UIKit

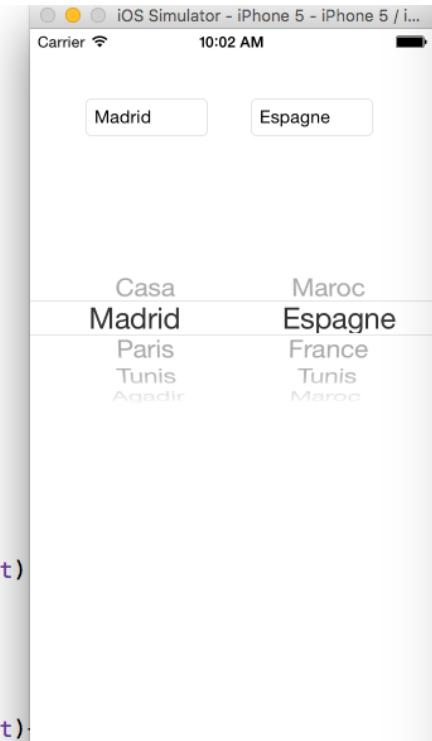
class ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
    @IBOutlet var txt : UITextField!
    @IBOutlet weak var txtspays: UITextField!
    @IBOutlet var picker : UIPickerView!

    var tableau = [[["Casa","Maroc"],["Madrid","Espagne"],["Paris","France"],["Tunis","Tunis"],
        ["Agadir","Maroc"],["Amsterdame","Belgique"],["Tarablouse","Lybie"]]

    // returns the number of 'columns' to display.
    func numberOfComponentsInPickerView(pickerView: UIPickerView) -> Int{
        return 2
    }

    // returns the # of rows in each component..
    func pickerView(pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int{
        return 7
    }
    func pickerView(pickerView: UIPickerView, titleForRow row: Int, forComponent component: Int) -> String!
    {
        return tableau[row][component]
    }

    func pickerView(pickerView: UIPickerView, didSelectRow row: Int, inComponent component: Int)
    {
        if component == 1 {
            txt.text = tableau[row][component]
        } else {txtspays.text = tableau[row][component]}
    }
    override func viewDidLoad() {
        super.viewDidLoad()
    }
}
```



Youssef  
BALOUKI

# Sélection d'une date dans un contrôle spécialisé : UIDatePicker

- \* Le contrôle Date Picker est très proche du contrôle Picker View,

```
// DatePicker
// Created by youssef balouki on 12/04/16.
// Copyright (c) 2016 youssef balouki. All rights reserved.
//

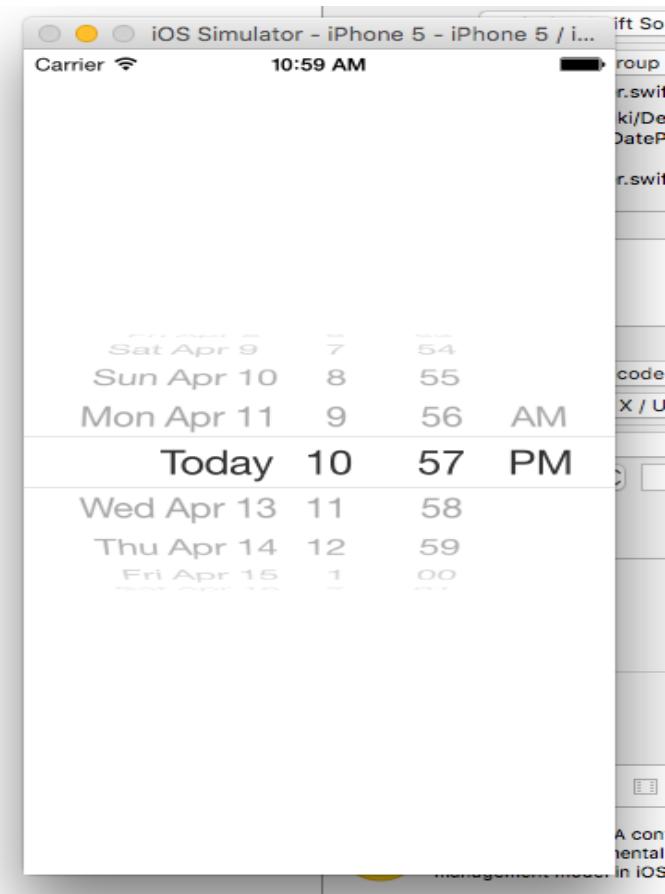
import UIKit

class ViewController: UIViewController {

    var datePicker = UIDatePicker()

    override func viewDidLoad() {
        super.viewDidLoad()
        datePicker.center = view.center
        view.addSubview(datePicker)
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



# Sélection d'une date dans un contrôle spécialisé : UIDatePicker

```
import UIKit

class ViewController: UIViewController {

    var datePicker = UIDatePicker()

    func dateChanged(datePicker_v : UIDatePicker){
        println("\(datePicker_v.date)")
    }
    override func viewDidLoad() {
        super.viewDidLoad()
        datePicker.center = view.center
        view.addSubview(datePicker)
        p_dpk.datePickerMode = .date
        datePicker.addTarget(self, action: "dateChanged:", forControlEvents: .ValueChanged)

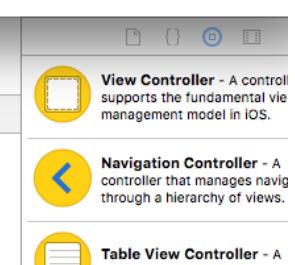
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



```
2016-04-13 10:32:42 +0000
2016-04-14 10:32:42 +0000
2016-04-14 11:32:42 +0000
```

Youssef  
BALOUKI



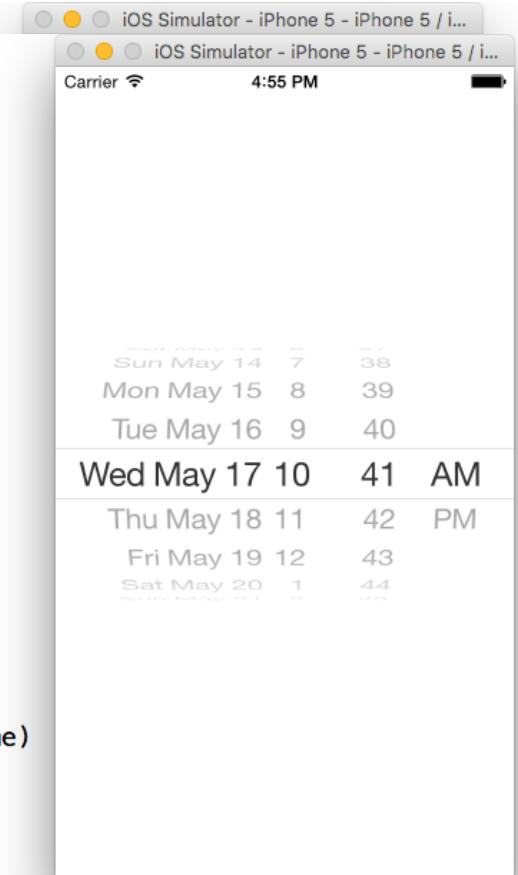
# Sélection d'une date dans un contrôle spécialisé : UIDatePicker

```
        }
    }
}
4
poi 4 */
import UIKit
as:
    class ViewController: UIViewController {
var
    var datePicker: UIDatePicker!
override
    override func viewDidLoad() {
        super.viewDidLoad()
        datePicker = UIDatePicker()
        datePicker.center = view.center
        view.addSubview(datePicker)

        let oneYearTime:NSTimeInterval = 365 * 24 * 60 * 60
        let todayDate = NSDate()

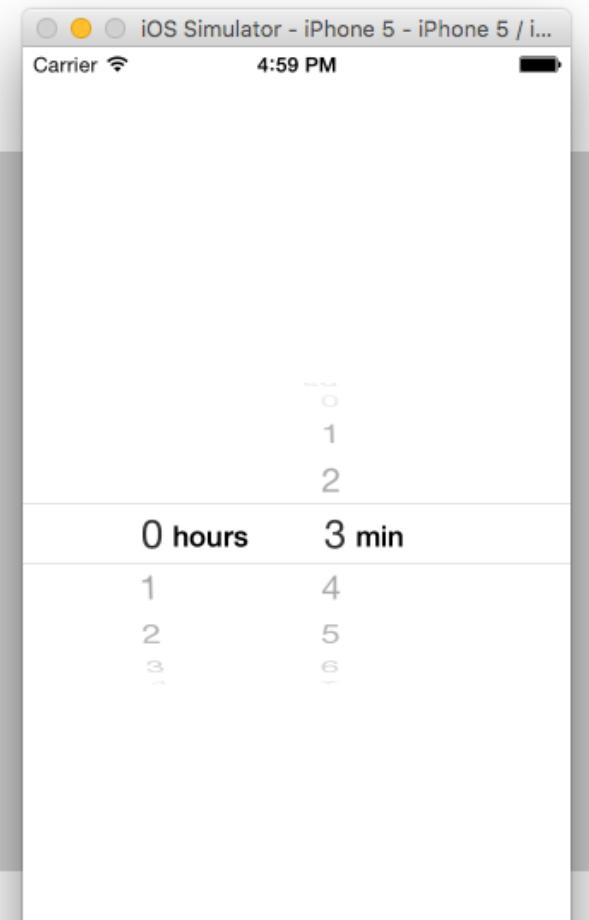
        let oneYearFromToday = todayDate.dateByAddingTimeInterval(oneYearTime)

        let twoYearsFromToday = todayDate.dateByAddingTimeInterval(2 * oneYearTime)
        datePicker.minimumDate = oneYearFromToday
        datePicker.maximumDate = twoYearsFromToday
    }
}
```



# Sélection d'une date dans un contrôle spécialisé : UIDatePicker

```
//  
// }  
//  
//}  
  
/* 5 */  
import UIKit  
  
class ViewController: UIViewController {  
  
    var datePicker: UIDatePicker!  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        datePicker = UIDatePicker()  
        datePicker.center = view.center  
        datePicker.datePickerMode = .CountDownTimer  
        let twoMinutes = (3 * 60) as NSTimeInterval  
        datePicker.countDownDuration = twoMinutes  
        view.addSubview(datePicker)  
  
    }  
  
}
```



# Recognizer Gesture

The screenshot shows the Xcode interface with a storyboard and a corresponding Swift code file.

**Storyboard:** On the left, a preview of an iPhone 7 screen displays a button with the text "bonjour" on top and "au revoir" on the bottom. The button has a blue rounded rectangular background.

**Code:** In the main editor, the `ViewController.swift` file contains the following code:

```
// ViewController.swift
// photos
//
// Created by Jaouad GFC on 03/05/2017.
// Copyright © 2017 Jaouad GFC. All rights reserved.

import UIKit

class ViewController: UIViewController {

    @IBAction func gesteSurImage(_ sender: UITapGestureRecognizer)
    {
        print("Gesture recognizer")
    }
}
```

**Simulator:** Below the code, a simulator window titled "iPhone 6 - iOS 10.2 (14C89)" shows the same button with the text "bonjour" and "au revoir".

**Assistant:** On the right, the Xcode Assistant pane is open, showing the "Tap Gesture Recognizer" under "Tap Gesture Recognizer - Recognizes tap gestures, including double-tap or multiple-touch." It also lists "Pinch Gesture Recognizer" and "Rotation Gesture Recognizer".

**Bottom Bar:** The Xcode toolbar at the bottom includes icons for View as: iPhone 7 (wC hR), zoom levels (55%, +), and file operations (Save, Open, Find, etc.).

# Recognizer Gesture

The screenshot shows the Xcode interface with a storyboard and a Swift code editor.

**Storyboard:** On the left, a preview of an iPhone screen displays a red rectangular button with the text "bonjour" and "au revoir".

**Code Editor:** The main area contains the following Swift code:

```
IBOutlet weak var photo: UIImageView!
@IBAction func act() {
    print("allo")
}

func toucher() {
    print("Image touchée !!!")
}

override func viewDidLoad() {
    super.viewDidLoad()

    let GesteSurImage = UITapGestureRecognizer(target: self, action: #selector(toucher))
    photo.addGestureRecognizer(GesteSurImage)
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning() // Dispose of any resources that can be recreated.
}
```

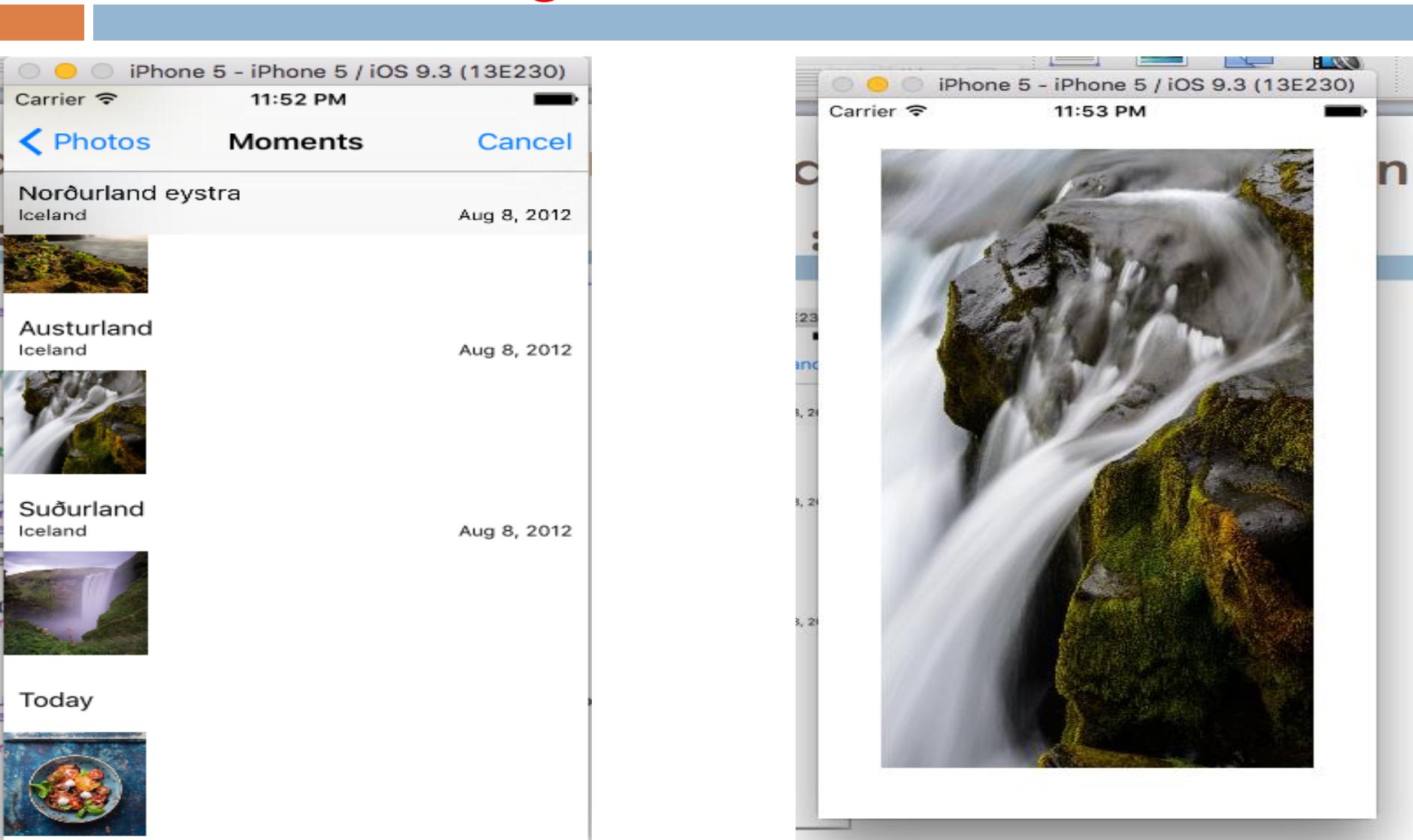
**Simulator:** On the right, a simulator window titled "iPhone 6 - iOS 10.2 (14)" shows the same red button with the text "bonjour" and "au revoir".

**Status Bar:** The status bar at the top of the simulator window shows "Carrier" and "3:26 PM".

**Bottom Bar:** The bottom of the Xcode interface has a toolbar with icons for file operations like Open, Save, and Find, along with a "photos" button.

**Text Output:** At the bottom left, the text "Image touchée !!!" is displayed, indicating a tap event was triggered.

# Sélection d'une image: **UIImagePickerController**



# Sélection d'une image : UIImagePickerController

```
class ViewController: UIViewController, UINavigationControllerDelegate, UIImagePickerControllerDelegate {
    @IBOutlet weak var Photo: UIImageView!

    @IBAction func selectPhoto(_ sender: UIGestureRecognizer) {
        //*** UIImagePickerController is a view controller that lets a user pick media from their photo library.
        let imageController = UIImagePickerController()

        //*** Only allow photos to be picked, not taken.
        imageController.sourceType = .photoLibrary
        imageController.allowsEditing = false

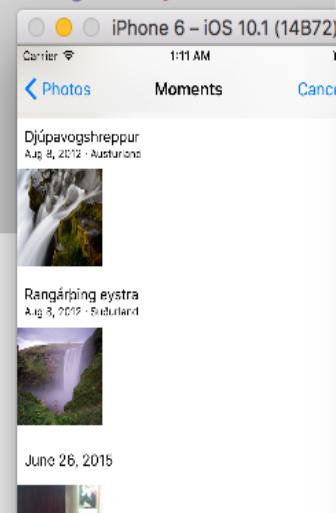
        //*** Make sure ViewController is notified when the user picks an image.
        imageController.delegate = self
        present(imageController, animated: true, completion: nil)
    }

    // ***The info dictionary may contain multiple representations of the image. You want to use the original.
    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
        //** Set photoImageView to display the selected image.
        Photo.image = info[UIImagePickerControllerOriginalImage] as! UIImage?

        //*** Dismiss the picker.
        dismiss(animated: true, completion: nil)
    }

    //*** Dismiss the picker if the user canceled.
    func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {
        dismiss(animated: true, completion: nil)
    }

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



# Configurer Info.plist.

- **une erreur peut se produire lorsque vous tentez de présenter le sélecteur d'image :**

Le système doit demander à l'utilisateur l'autorisation avant d'accéder à sa photothèque. Dans iOS 10 et versions ultérieures, vous devez fournir une description d'utilisation de la bibliothèque photo. Cette description explique pourquoi votre application souhaite accéder à la bibliothèque photo.

# Configurer info.plist

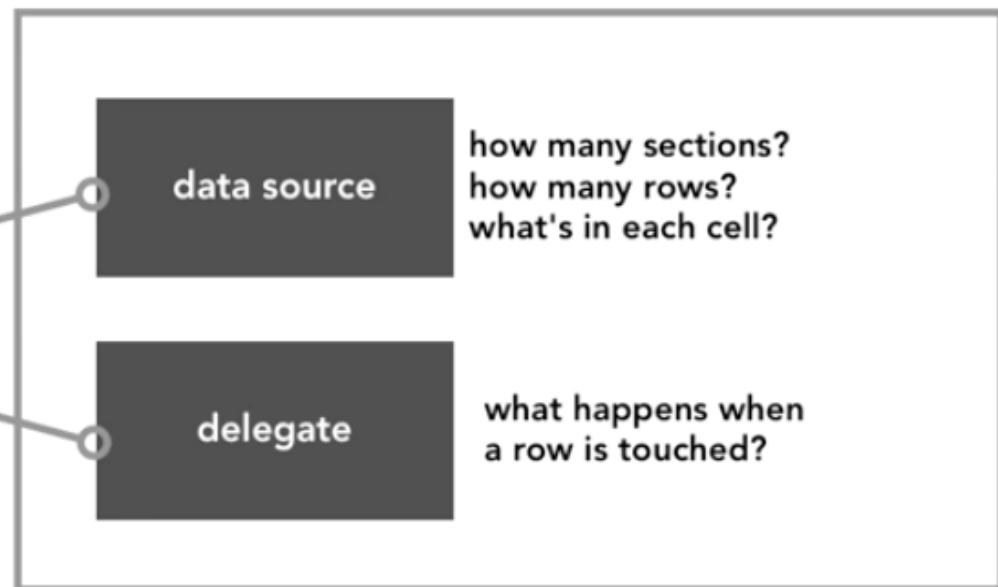
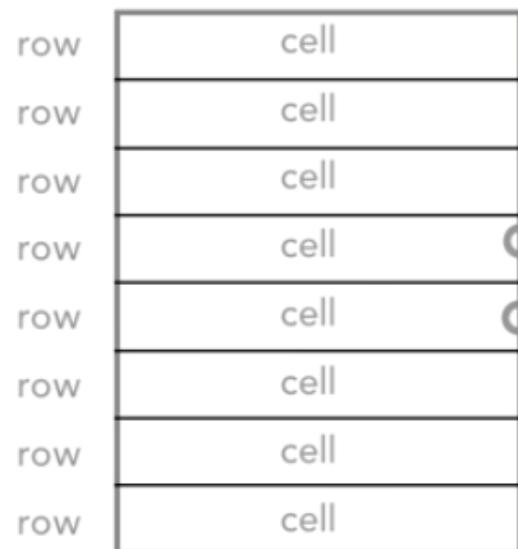
- Pour ajouter une description d'utilisation de la bibliothèque photo :
  1. Dans le navigateur du projet, sélectionnez Info.plist.
  2. Pour ajouter un nouvel élément, passez le dessus du dernier élément dans la liste des propriétés et cliquez sur le bouton Ajouter lorsqu'il apparaît (ou sélectionnez Éditeur> Ajouter un élément).
  3. Dans le menu contextuel, faites défiler vers le bas et choisissez **Privacy - Photo Library Usage Description**.
  4. Dans la nouvelle ligne, assurez-vous que Type est défini String. Ensuite, double-cliquez sur la section Valeur et saisissez un message

# TableView & TableView Cell

Icône	Nom du contrôle	Fonction
	Table View	Liste hiérarchique d'informations textuelles disposées verticalement
	Table View Cell	Paramètre d'une des cellules affichées dans un Table View

# Table View

## Table View concepts



ViewController

# Table view

```
// Created by yousset balouki on 3/06/16.
// Copyright © 2016 yousset balouki. All rights reserved.
//

import UIKit

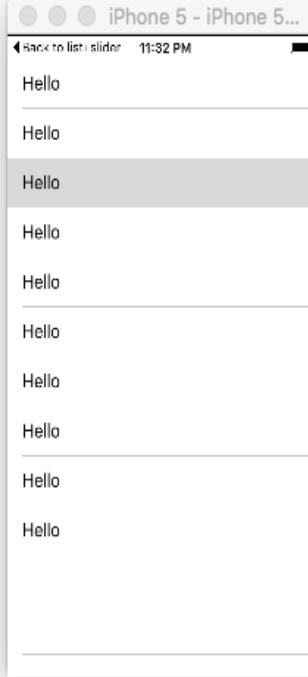
class ViewController: UIViewController, UITableViewDataSource, UITableViewDelegate{

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return 10
    }

    func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = UITableViewCell()
        cell.textLabel!.text = "Hello"
        return cell
    }
}
```



Youssef  
BALOUKI

# La réutilisation des cellules de table

Problème : Créer trop cellules

one
two
three
four
five
six
seven
eight
nine
ten
eleven
twelve
thirteen
fourteen
fifteen

# La réutilisation des cellules de table

Solution : Réutiliser les cellules

dequeued cell
two
three
four
five
six
seven
eight
nine

# Loading a list into a Table View

```
import UIKit

class ViewController: UIViewController, UITableViewDataSource, UITableViewDelegate{

    let cours = [("iOS App dev","balouki youssef"),("langage Swift","Y.Balouki"),
        ("Android","Chegdali"),("langage JAvA","Y.balouki"),("POO","Chakib"),("langage
        UML","Hassani"),("Langage c","H.Essoufi")]

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    func numberOfSectionsInTableView(tableView: UITableView) -> Int {
        return 1
    }
    func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return cours.count
    }

    func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) ->
        UITableViewCell {
        // let cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
        let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)
        let (modules, prof) = cours[indexPath.row]
        cell.textLabel!.text = modules
        return cell
    }
}
```

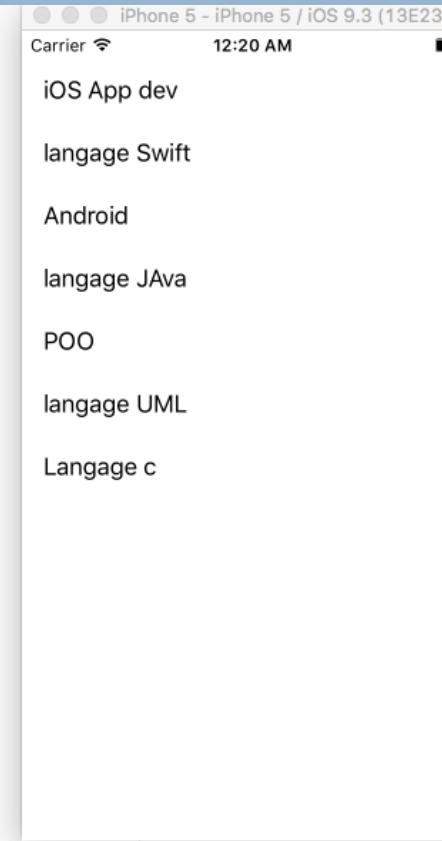


Table View Controller - A controller that manages a table



Table View - Displays data in a plain, sectioned, or grouped



Table View Cell - Defines the attributes and behavior of cells in a table view.

⚠️ Immutable value 'prof' was never used; consider replacing with '\_' or removing it

# Personnaliser Table View

```
class ViewController: UIViewController, UITableViewDataSource, UITableViewDelegate{

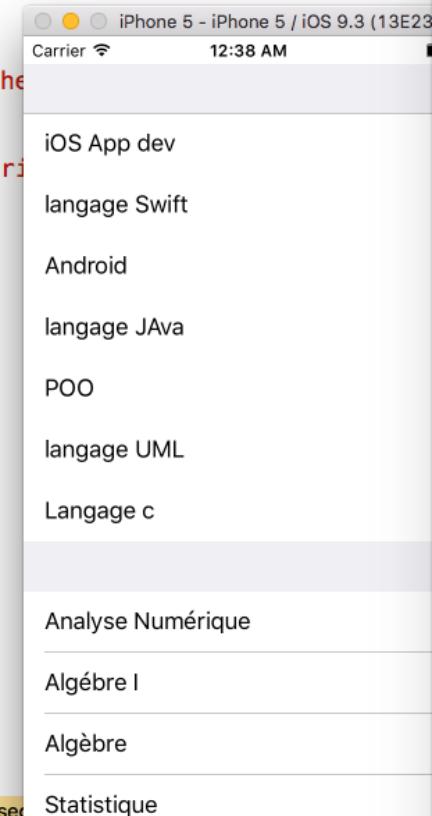
    let coursInfo = [("iOS App dev","balouki youssef"),("langage Swift","Y.Balouki"),("Android","Che
        JJava","Y.balouki"),("POO","Chakib"),("langage UML","Hassani"),("Langage c","H.Essoufi")]

    let coursMath = [("Analyse Numérique","H.KHALISS"),("Algébre I","Y.lamnii"),("Algèbre","M.zakari
        ("Statistique","B.Yaakobi")]

        func numberOfSectionsInTableView(tableView: UITableView) -> Int {
    return 2
}
func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    if section == 0 {
        return coursInfo.count
    } else {
        return coursMath.count
    }
}

func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath)
-> UITableViewCell {
// let cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
    let cell = tableView.dequeueReusableCell(withIdentifier: "cell", forIndexPath: indexPath)

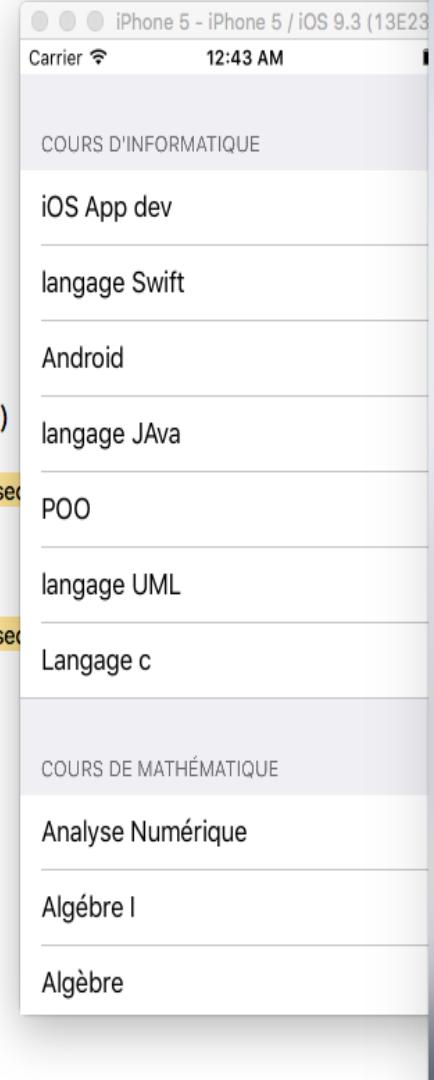
    if indexPath.section == 0 {
        let (modules, prof) = coursInfo[indexPath.row] ⚠️ Immutable value 'prof' was never used
        cell.textLabel!.text = modules
    } else {
        let (modules, prof) = coursMath[indexPath.row] ⚠️ Immutable value 'prof' was never used
        cell.textLabel!.text = modules
    }
    return cell
}
override func viewDidLoad() {
```



Youssef  
BALOUKI

# Loading a list into a Table View

```
func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {  
    if section == 0 {  
        return coursInfo.count }  
    else {  
        return coursMath.count }  
}  
  
func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {  
    // let cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")  
    let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)  
    if indexPath.section == 0 {  
        let (modules, prof) = coursInfo[indexPath.row] ⚠ Immutable value 'prof' was never used  
        cell.textLabel!.text = modules}  
  
    else {  
        let (modules, prof) = coursMath[indexPath.row] ⚠ Immutable value 'prof' was never used  
        cell.textLabel!.text = modules}  
    return cell  
}  
  
func tableView(tableView: UITableView, titleForHeaderInSection section: Int) -> String? {  
    if section == 0 {  
        return "Cours D'informatique"  
    } else {  
        return "Cours De MATHématique"  
    }  
}
```



Youssef  
BALOUKI

# Personnaliser Table View

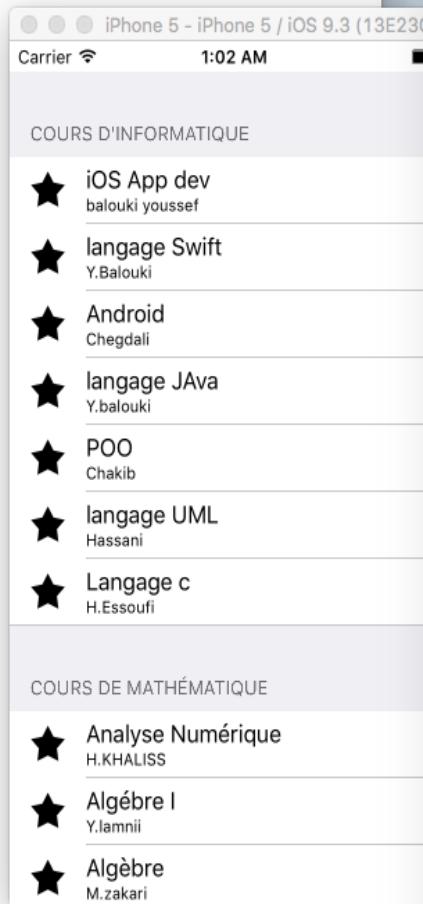
```
func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    // let cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
    let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)
    if indexPath.section == 0 {
        let (modules, prof) = coursInfo[indexPath.row]
        cell.textLabel!.text = modules
        cell.detailTextLabel?.text = prof
    }
    else {
        let (modules, prof) = coursMath[indexPath.row]
        cell.textLabel!.text = modules
        cell.detailTextLabel?.text = prof} // subtitle
    return cell
}

func tableView(tableView: UITableView, titleForHeaderInSection section: Int) -> String? {
    if section == 0 {
        return "Cours D'informatique"
    } else {
        return "Cours De MATHématique"
    }
}
override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view, typically from a nib.
}
```

Carrier	iPhone 5 - iPhone 5 / iOS 9.3 (13E230)	12:51 AM
COURS D'INFORMATIQUE		
iOS App dev	balouki youssef	
langage Swift	Y.Balouki	
Android	Chegdaï	
langage JAvA	Y.balouki	
POO	Chakib	
langage UML	Hassani	
Langage c	H.Essoufi	
COURS DE MATHÉMATIQUE		
Analyse Numérique	H.KHALISS	
Algébre I	Y.Iamnii	
Algèbre	M.zakari	

# Personnaliser Table View

```
func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {  
    // let cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")  
    let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)  
    if indexPath.section == 0 {  
        let (modules, prof) = coursInfo[indexPath.row]  
        cell.textLabel!.text = modules  
        cell.detailTextLabel?.text = prof  
    }  
  
    else {  
        let (modules, prof) = coursMath[indexPath.row]  
        cell.textLabel!.text = modules  
        cell.detailTextLabel?.text = prof} // subtitle  
  
    let newImage = UIImage(named: "star") // afficher image ds cellule  
    cell.imageView?.image = newImage  
    return cell  
}  
  
func tableView(tableView: UITableView, titleForHeaderInSection section: Int) -> String? {  
    if section == 0 {  
        return "Cours D'informatique"  
    } else {  
        return "Cours De MATHématique"  
    }  
}  
override func viewDidLoad() {  
    super.viewDidLoad()  
    // Do any additional setup after loading the view, typically from a nib.  
}
```



iPhone 5 - iPhone 5 / iOS 9.3 (13E230)

Carrier 1:02 AM

COURS D'INFORMATIQUE

- ★ iOS App dev  
balouki youssef
- ★ langage Swift  
Y.Balouki
- ★ Android  
Chedgali
- ★ langage JAvA  
Y.balouki
- ★ POO  
Chakib
- ★ langage UML  
Hassani
- ★ Langage c  
H.Essoufi

COURS DE MATHÉMATIQUE

- ★ Analyse Numérique  
H.KHALISS
- ★ Algébre I  
Y.Iamnii
- ★ Algèbre  
M.zakari

Youssef  
BALOUKI

# TP :

```
class ViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {

    @IBOutlet weak var tbl: UITableView!
    @IBOutlet weak var sld: UISlider!
    @IBAction func sld(sender: AnyObject) {
        tbl.reloadData()
    }

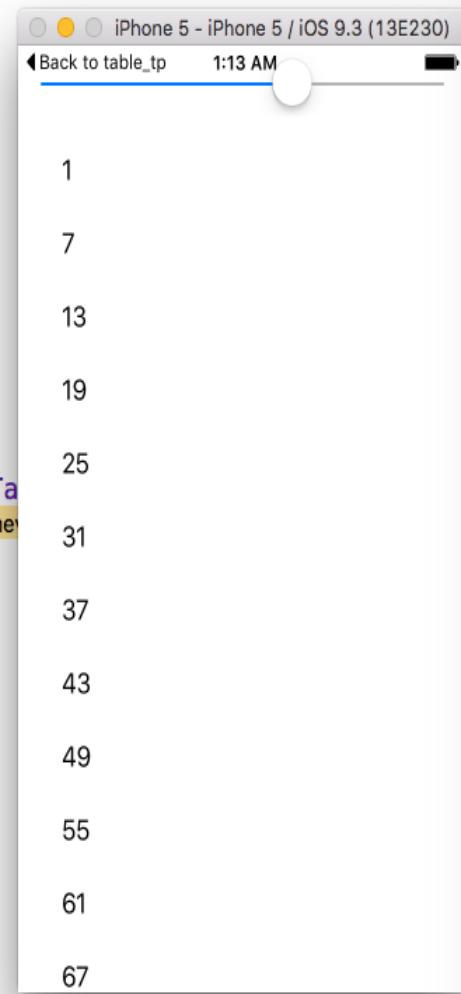
    func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return 20
    }

    func tableView(tableView: UITableView, cellForRowAt indexPath: NSIndexPath) -> UITableViewCell {
        var cell = UITableViewCell()
        cell.textLabel?.text = "\((Int(sld.value*10) * indexPath.row + 1))"

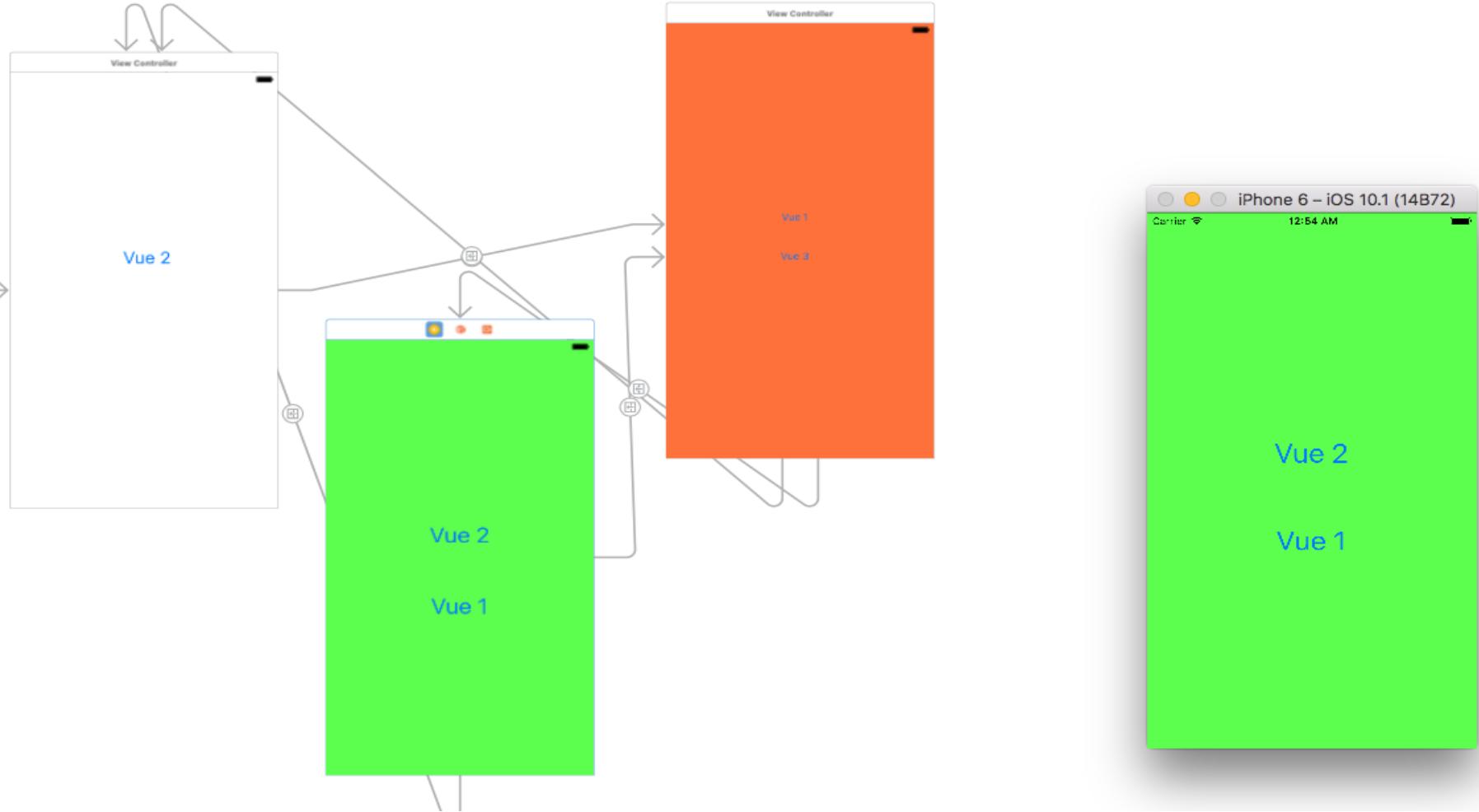
        return cell
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

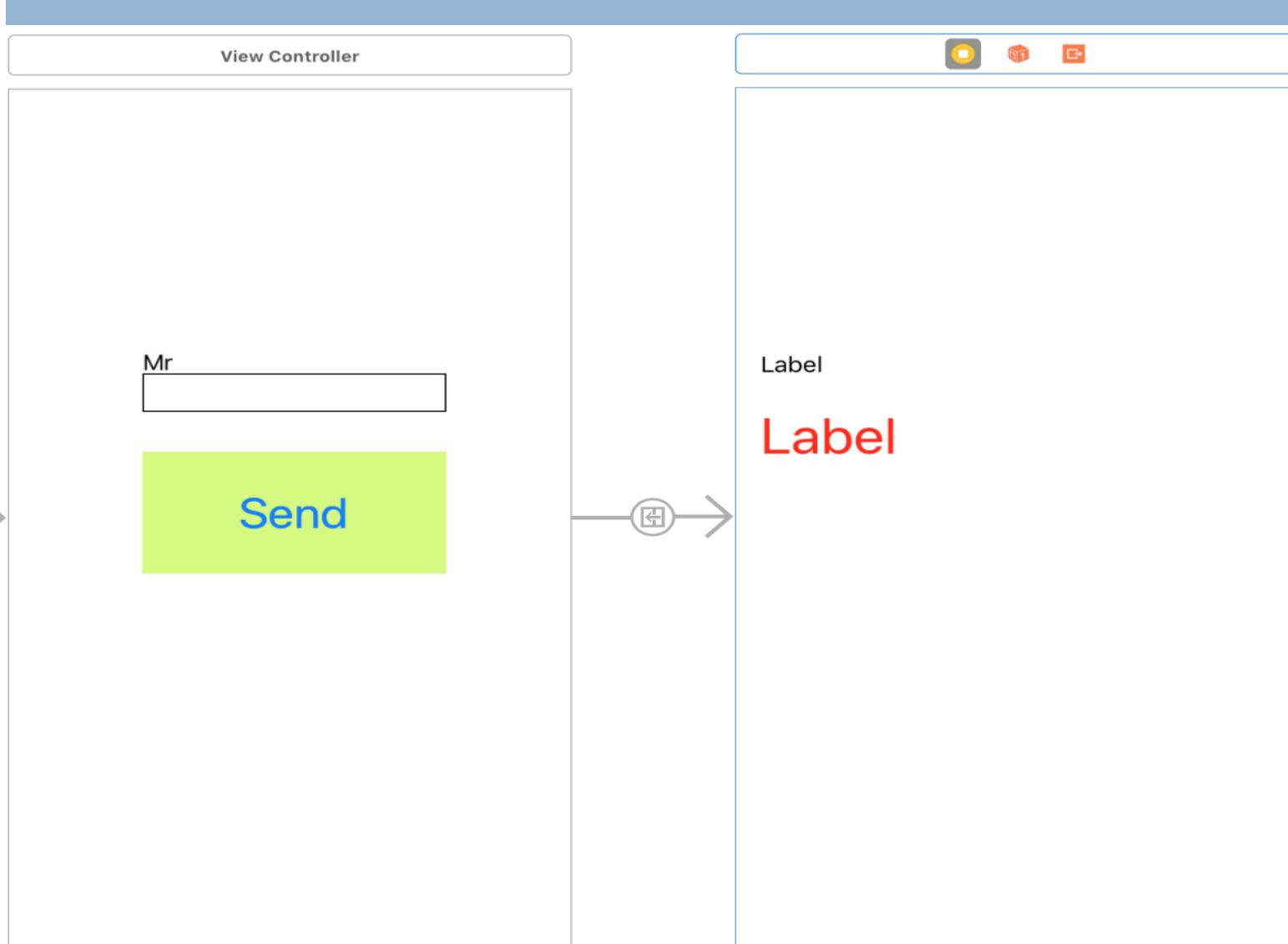
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



# Navigation : Segue



# Segue Personnalisé

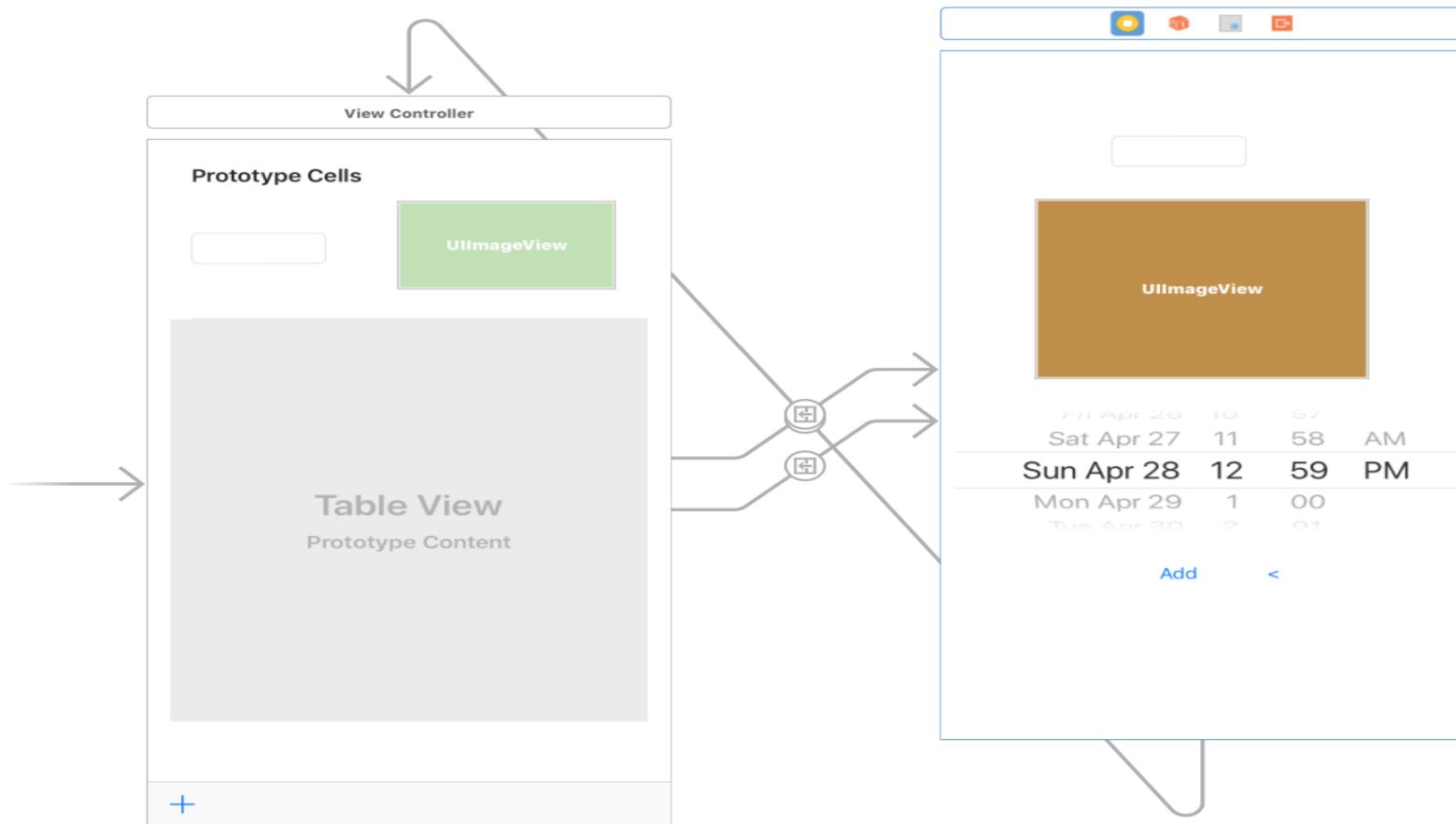


# Segue Personnalisé

```
class ViewController: UIViewController {  
  
    @IBOutlet weak var person: UITextField!  
  
    override func didReceiveMemoryWarning() {  
        super.didReceiveMemoryWarning()  
    }  
  
    override func prepare(for segue: UIStoryboardSegue,  
                         sender: Any?) {  
        if segue.identifier == "sg" {  
            let ss = segue.destination as! ViewController2  
            ss.message = "Bonjour"  
            DispatchQueue.main.async {  
                ss.person.text = self.person.text  
            }  
        }  
    }  
}
```

```
class ViewController2: UIViewController {  
  
    var message : String = ""  
  
    @IBOutlet weak var msg: UILabel!  
    @IBOutlet weak var person: UILabel!  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        msg.text = message  
    }  
}
```

# Segue Personnalisé : Sélection dans TableView



# Segue Personnalisé : Sélection dans TableView

```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    let indice = indexPath.row // envoie de l'indice choisi
    self.performSegue(withIdentifier: "mySegue", sender: indice)
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "mySegue" {

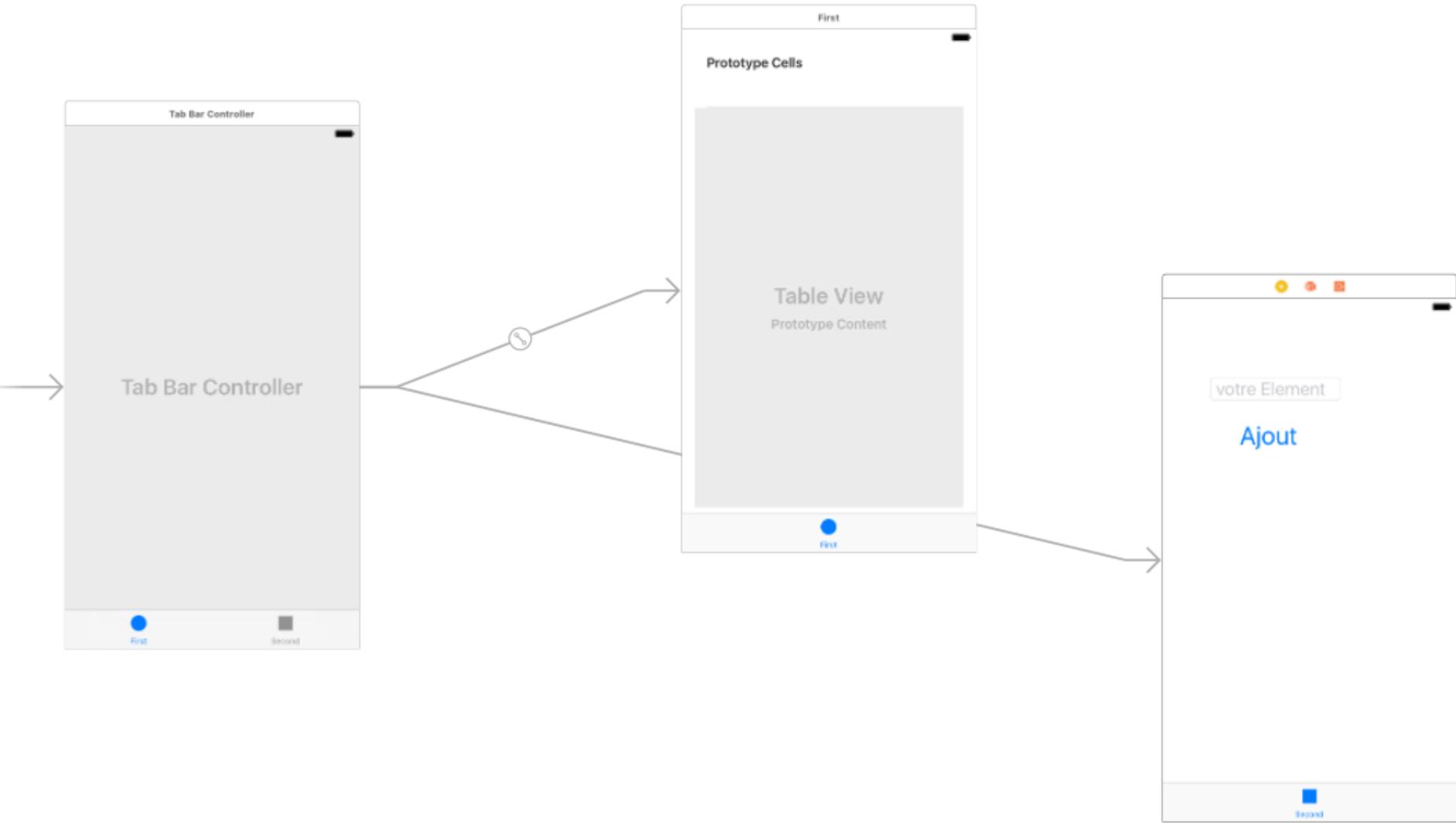
        let cv = segue.destination as! ViewController2
        cv.person = tab[sender as! Int] // envoie de la personne ayant l'indice : sender
        cv.indice = sender as! Int
    }

    if segue.identifier == "addsg" {
        let cv = segue.destination as! ViewController2
        cv.indice = -1
    }
}
```

# Segue Personnalisé : Sélection dans TableView

```
class ViewController2: UIViewController, UINavigationDelegate, UIImagePickerControllerDelegate {  
  
    var person = Person()  
    var indice = 0  
  
    @IBOutlet weak var nom: UITextField!  
    @IBOutlet weak var img: UIImageView!  
    @IBOutlet weak var pikdate: UIDatePicker!  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        if indice != -1 {  
            nom.text = person.nom  
            img.image = UIImage(data: person.photo as! Data)  
        }  
    }  
}
```

# Tab Application



# Une vue secondaire pour l'ajout

```
import UIKit

var toDoList = [String]()
class FirstViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {

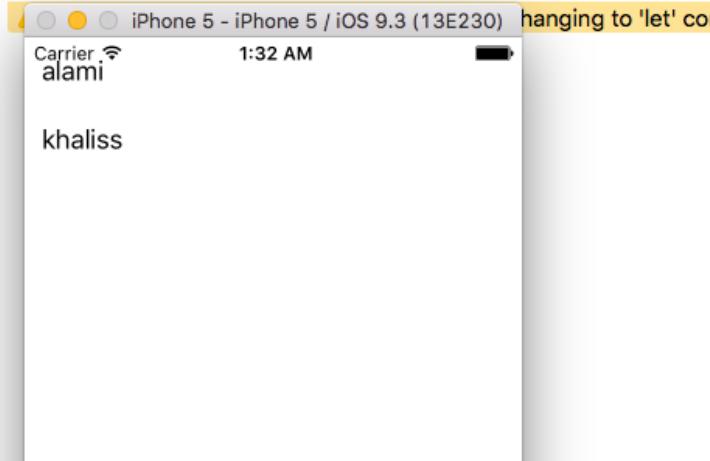
    @IBOutlet weak var tbl: UITableView!
    override func viewDidAppear(animated: Bool) {
        tbl.reloadData()
    }

    func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return toDoList.count
    }

    func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) -> UITableViewCell {
        var cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
        cell.textLabel!.text = toDoList[indexPath.row]
        return cell
    }

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

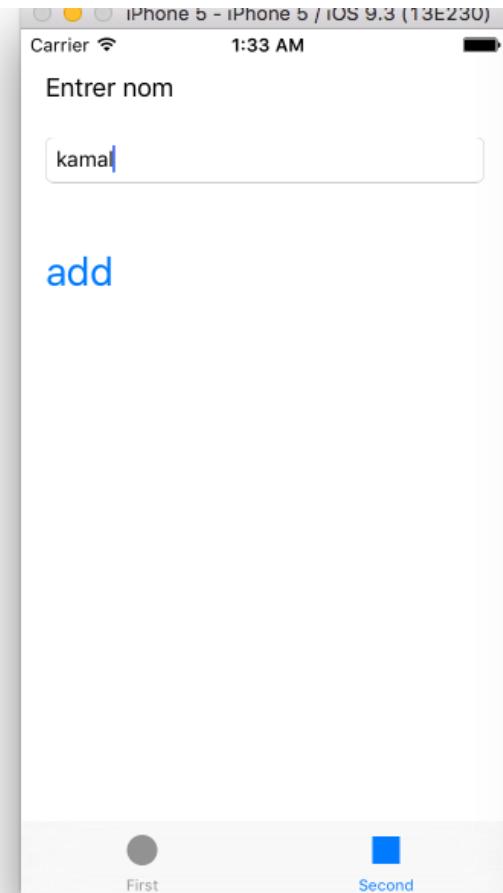
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }
}
```



The screenshot shows the iPhone 5 simulator running iOS 9.3. The table view displays two items: 'alami' and 'khaliss'. The status bar indicates the carrier is 'Carrier', the time is '1:32 AM', and there is a signal strength icon.

# Une vue secondaire pour l'ajout

```
class SecondViewController: UIViewController, UITextFieldDelegate {  
  
    @IBOutlet weak var lbl: UILabel!  
    @IBOutlet weak var txt: UITextField!  
    @IBAction func add(sender: AnyObject) {  
  
        toDoList.append(txt.text!)  
        txt.text = ""  
    }  
  
    func textFieldDidEndEditing(textField: UITextField) {  
        textField.resignFirstResponder()  
    }  
    func textFieldShouldReturn(textField: UITextField) -> Bool {  
        textField.resignFirstResponder()  
        return true  
    }  
    override func touchesBegan(touches: Set<UITouch>, withEvent event: UIEvent?) {  
        self.view.endEditing(true)  
    }  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        // Do any additional setup after loading the view, typically from a nib.  
    }  
  
    override func didReceiveMemoryWarning() {  
        super.didReceiveMemoryWarning()  
        // Dispose of any resources that can be recreated.  
    }  
}
```



# Loading a list & Delete Row

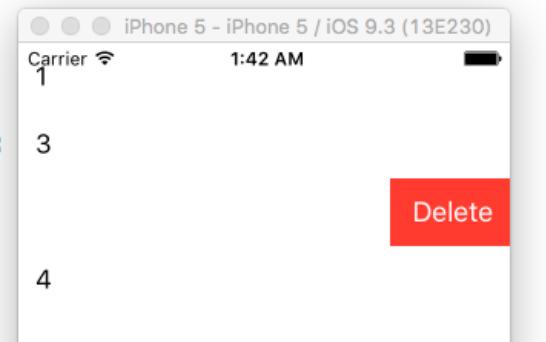
```
override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
}

override func viewDidAppear(animated: Bool) {
    tbl.reloadData()
}

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return ToDoList.count
}

func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) -> UITableViewCell {
    var cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
    cell.textLabel!.text = ToDoList[indexPath.row]
    ⚠ Variable 'cell' was never mutated; consider changing to 'let' constant
    return cell
}

func tableView(tableView: UITableView, commitEditingStyle editingStyle: UITableViewCellEditingStyle, forRowAtIndexPath indexPath: NSIndexPath) {
    if editingStyle == UITableViewCellEditingStyle.Delete {
        ToDoList.removeAtIndex(indexPath.row)
        NSUserDefaults.standardUserDefaults().setObject(ToDoList, forKey: "ToDoList")
        tbl.reloadData()
    }
}
```



Youssef  
BALOUIK

# Stockage Permanent :**UserDefault**s :

- ✓ **UserDefault**s est un excellent moyen d'enregistrer rapidement et facilement des données persistantes pendant la durée de vie de l'application.
- ✓ Ces variables sont enregistrées tant que l'application n'a pas été supprimée.
- ✓ Exemple, vous pouvez demander le nom d'un utilisateur, puis le sauvegarder, chaque fois qu'ils ouvrent l'application, le nom apparaît.

# Stockage Permanent:**UserDefaults**

- **un dictionnaire de** de type **[String : Any]** qui persiste  
Vous définissez des clefs de type String et vous pouvez y stocker des valeurs. Pour récupérer la valeur, vous utilisez la **clef**.
- Les valeurs peuvent être une combinaison des types suivants :
  - **Array**
  - **Dictionary**
  - **String**
  - **Date**
  - **Data**
  - **Nombre (Int, Double, Float)**

# Stockage Permanent :**UserDefault**s :

```
// Fonction pour écrire des données  
set(Any? , forKey: String)
```

\*\*\* **Any** doit être une Property List, sinon crash.

```
// fonction pour lire des données  
object(forKey: String) -> Any?
```

# Stockage Permanent

```
override func viewDidLoad() {
    super.viewDidLoad()

    let defaults = UserDefaults.standard
    // Stockage des différentes valeurs

    defaults.set(15, forKey: "longueur")
    defaults.set(true, forKey: "actif")
    defaults.set(25.8, forKey: "distance")
    defaults.set("Rachid", forKey: "nom")
    defaults.set(Date(), forKey: "date")
    let prenomsParents = ["Ahmed", "Fatima"]
    defaults.set(prenomsParents, forKey: "parents")

    // lectures à partir d' UserDefaults

    let weightValue = defaults.integer(forKey: "longueur");           print("la longueur = \(weightValue)")
    let estActif = defaults.bool(forKey: "actif");                      print("actif (oui/non) = \(estActif)")
    let distanceValue = defaults.double(forKey: "distance");          print("distance = \(distanceValue)")
    let nom = defaults.object(forKey: "nom") as! String;               print("Nom = \(nom)")
    let date = defaults.object(forKey: "date") as! Date;                print(" date courante = \(date)")
    let prenomParents = defaults.object(forKey: "parents") as! [String]
    print("Prenoms parents : \(prenomParents[0]) et \(prenomParents[1])")
```

PermanentStorage

```
la longueur = 15
actif (oui/non) = true
distance = 25.8
Nom = Rachid
date courante = 2017-04-19 23:31:04 +0000
Prenoms parents : Ahmed et Fatima
```

# Stockage Permanent

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        UserDefaults.standard.set("balouki", forKey: "nom")
        let monNom = UserDefaults.standard.object(forKey: "nom")!

        print(monNom)
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

# Premier Contrôleur : UserDefaults

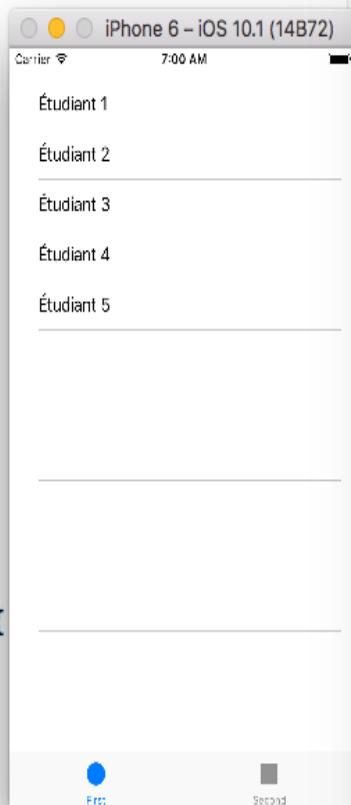
```
import UIKit
var liste = [String]()
class FirstViewController: UIViewController, UITableViewDelegate, UITableViewDataSource {

    @IBOutlet weak var maTable: UITableView!
    override func viewDidLoad() {
        super.viewDidLoad()
        if UserDefaults.standard.object(forKey: "maliste2") != nil {
            liste = UserDefaults.standard.object(forKey: "maliste2") as! [String]
        }
    }
    func numberOfSections(in tableView: UITableView) -> Int {
        return 1
    }
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return liste.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cl = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)

        cl.textLabel?.text = liste[indexPath.row]
        return cl
    }

    override func viewDidAppear(_ animated: Bool) {
        maTable.reloadData()
    }
    func tableView(_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {
        if editingStyle == UITableViewCellEditingStyle.delete {
            liste.remove(at: indexPath.row)
            UserDefaults.standard.set(liste, forKey: "maliste2")
            maTable.reloadData()
        }
    }
    override func didReceiveMemoryWarning() {
```



# Deuxième Contrôleur

```
import UIKit

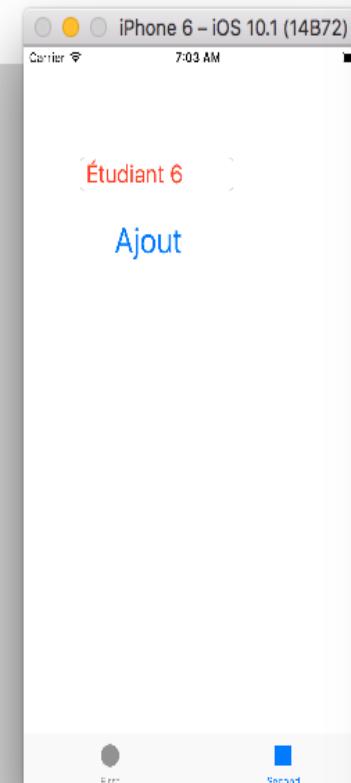
class SecondViewController: UIViewController, UITextFieldDelegate {
    @IBOutlet weak var txt: UITextField!

    @IBAction func add(_ sender: UIButton) {
        liste.append(txt.text!)
        UserDefaults.standard.set(liste, forKey:"maliste2")
        txt.text = ""
    }

    override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
        self.view.endEditing(true)
    }

    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        txt.resignFirstResponder()
        return true
    }
    override func viewDidLoad() {
        super.viewDidLoad()
    }

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



# Loading a list into a Table View

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

override func viewDidAppear(animated: Bool) {
    tbl.reloadData()
}

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return toDoList.count
}

func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    var cell = UITableViewCell(style: .Default, reuseIdentifier: "cell")
    cell.textLabel!.text = toDoList[indexPath.row]
    △ Variable 'cell' was never mutated; consider changing to 'let' constant
    return cell
}

func tableView(tableView: UITableView, commitEditingStyle editingStyle: UITableViewCellEditingStyle,
forRowAtIndexPath indexPath: IndexPath){
    if editingStyle == UITableViewCellEditingStyle.Delete {
        toDoList.removeAtIndex(indexPath.row)
        UserDefaults.standard.setObject(toDoList, forKey: "toDoList")
        tbl.reloadData()
    }
}
```

Youssef  
BALOUKI



# Costomized Cell

The screenshot shows the Xcode interface with the following details:

- Toolbar:** Standard Xcode toolbar with icons for file, search, and navigation.
- Project Navigator:** Shows a project named "table". Inside the "table" folder, there are files: AppDelegate.swift, ViewController.swift, Main.storyboard, Assets.xcassets, LaunchScreen.storyboard, Info.plist, and notreNouveauTableViewCell.swift (which is selected).
- Editor:** Displays the code for notreNouveauTableViewCell.swift. The code is as follows:

```
// notreNouveauTableViewCell.swift
// table
//
// Created by youssef balouki on 14/01/17.
// Copyright © 2017 youssef balouki. All rights reserved.

import UIKit

class notreNouveauTableViewCell: UITableViewCell {

    @IBOutlet weak var solde: UILabel!
    @IBOutlet weak var pays: UILabel!
    @IBOutlet weak var capitale: UILabel!

    @IBOutlet weak var img: UIImageView!
}
```

# Image Picker

```
import UIKit
//var agent = [(String,String,AnyObject)]()
var agent = [String]()
var agentPrenom = [String]()
var imgs = [AnyObject]()

class FirstViewController: UIViewController,UITableViewDelegate,UITableViewDataSource {

@IBOutlet weak var tbl: UITableView!
override func viewDidLoad() {
    super.viewDidLoad()

    if UserDefaults.standard.object(forKey: "myListNoms") != nil {
        agent = UserDefaults.standard.object(forKey: "myListNoms") as! [String]
        agentPrenom = UserDefaults.standard.object(forKey: "myListPrenoms") as! [String]
        imgs = UserDefaults.standard.object(forKey: "myListPhotos") as! [AnyObject]
    }
}

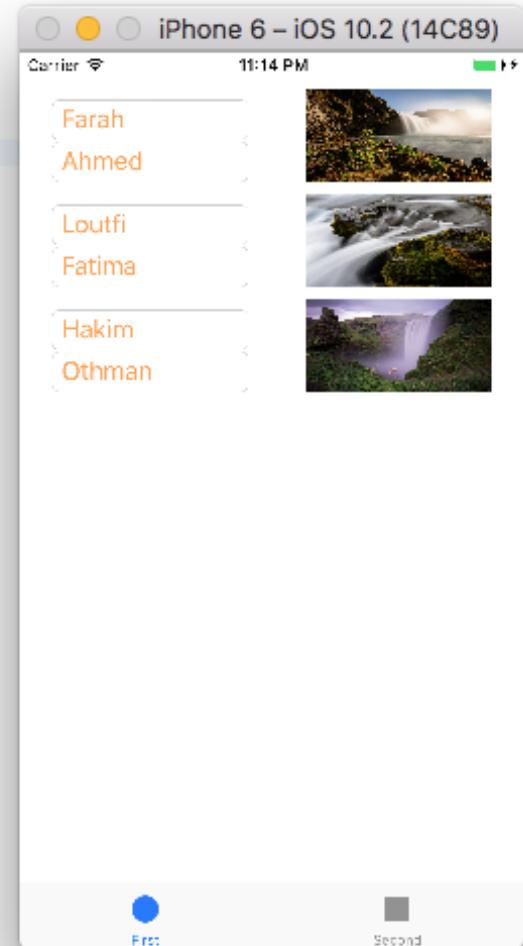
func numberOfSections(in tableView: UITableView) -> Int {
    return 1
}

func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return agent.count
}

override func viewDidAppear(_ animated: Bool) {    tbl.reloadData() }

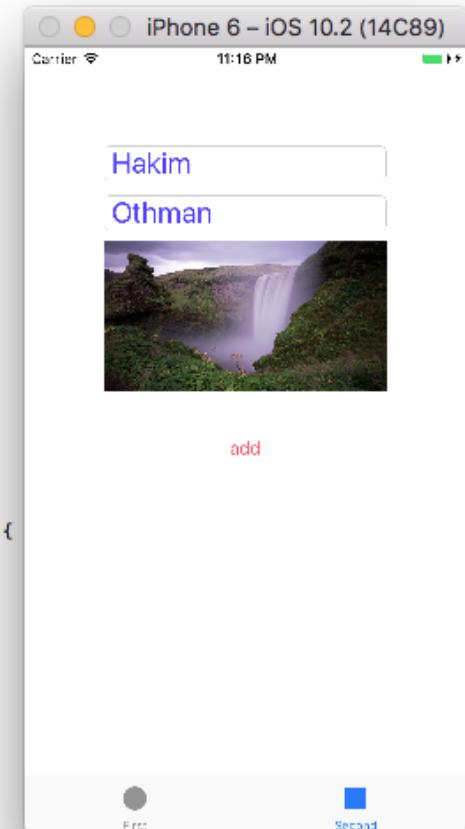
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cellule = tableView.dequeueReusableCell(withIdentifier:"cell", for: indexPath) as! MyCell
    cellule.nom.text = agent[indexPath.row]
    cellule.prenom.text = agentPrenom[indexPath.row]
    cellule.photo.image = UIImage(data :imgs[indexPath.row] as! Data )

    return cellule
}
override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Dispose of any resources that can be recreated.
}
```



# Image Picker

```
import UIKit
class SecondViewController: UIViewController, UINavigationControllerDelegate, UIImagePickerControllerDelegate, UITextFieldDelegate {
    @IBOutlet weak var nom: UITextField!
    @IBOutlet weak var prenom: UITextField!
    @IBOutlet weak var photo: UIImageView!
    @IBAction func add(_ sender: UIButton) {
        let v = UIImagePNGRepresentation(photo.image!)
        agent.append(nom.text!)
        agentPrenom.append(prenom.text!)
        imgs.append(v! as AnyObject)
        UserDefaults.standard.set(agent, forKey: "myListNoms")
        UserDefaults.standard.set(agentPrenom, forKey: "myListPrenoms")
        UserDefaults.standard.set(imgs, forKey: "myListPhotos")
    }
    @IBAction func geste(_ sender: UITapGestureRecognizer) {
        let myPick = UIImagePickerController()
        myPick.sourceType = .savedPhotosAlbum
        myPick.delegate = self
        present(myPick, animated: true, completion: nil)
    }
    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
        self.dismiss(animated: true, completion: nil)
        photo.image = info[UIImagePickerControllerOriginalImage] as! UIImage
    }
    override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
        view.endEditing(true)
    }
    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
    }
    override func viewDidLoad() {
        super.viewDidLoad()
    }
}
```



# TableView avec UIImagePickerController

```
import UIKit// Copyright © 2017 youssef balouki. All rights
class ViewController: UIViewController, UITableViewDataSource, UITableViewDelegate{

var tab = ["Martin", "Bernard", "Dubois", "Simon", "Garcia", "Fourier", "Nicolas"]
var tab1 = ["Albert", "Natacha", "Mari", "Francois", "Pierre", "Jack", "Michel"]
var tab4 = ["01315 200362 001", "01315 200365 001", "01315 200464 001", "01315 200300 001", "01315
001", "01315 200856 001"]
var tab2 = ["client1", "client2", "client3", "client4", "client1", "empty", "empty"]
var tab3 = ["1233.34", "50 000.00", "23.45", "7000.00", "300.00", "1256.99", "99.99"]

@IBOutlet weak var tbl: UITableView!
func numberOfSections(in tableView: UITableView) -> Int
{return 1}

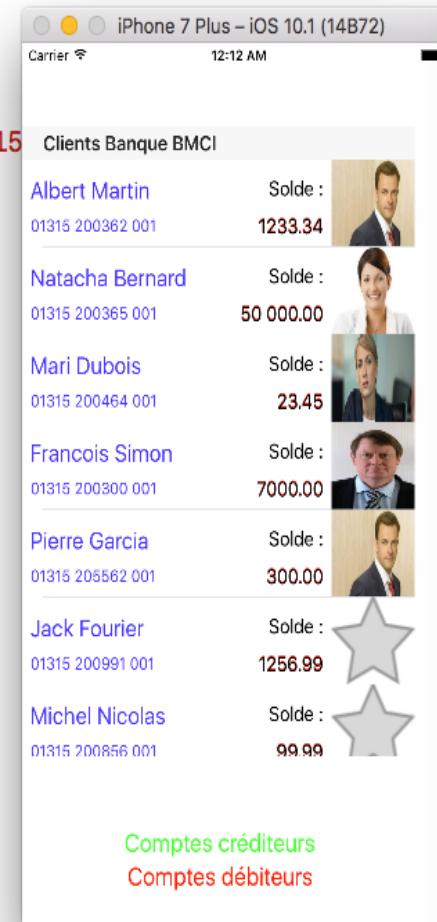
func tableView(_ tableView: UITableView, titleForHeaderInSection section: Int) -> String? {
    return "Clients Banque BMCI "
}

@IBAction func act(_ sender: UIButton) {
    tab = ["Charrière", "Bernard", "Dubois", "Simon", "Garcia", "Fourier", "Nicolas"]
    tab1 = ["Albert", "Natacha", "Mari", "Francois", "Pierre", "Jack", "Michel"]
    tab2 = ["star", "star", "star", "empty", "empty", "empty", "star"]

    self.tbl.reloadData()
}

func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int
{return tab.count}

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
let v = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath) as! notreNouveauTableViewCell
v.pays.text = tab4[indexPath.row]
v.capitale.text = "\(tab1[indexPath.row]) \(tab[indexPath.row])"
v.img.image = UIImage(named: tab2[indexPath.row])
v.solde.text = tab3[indexPath.row]
return v
}
```



# Mode Programmation

1. Ajouter des # vues par programmation
2. Mettre en forme des # vues par programation
3. Creation des actions sur les # vues par programmation

# Ajout de boutons à l'interface utilisateur avec UIButton

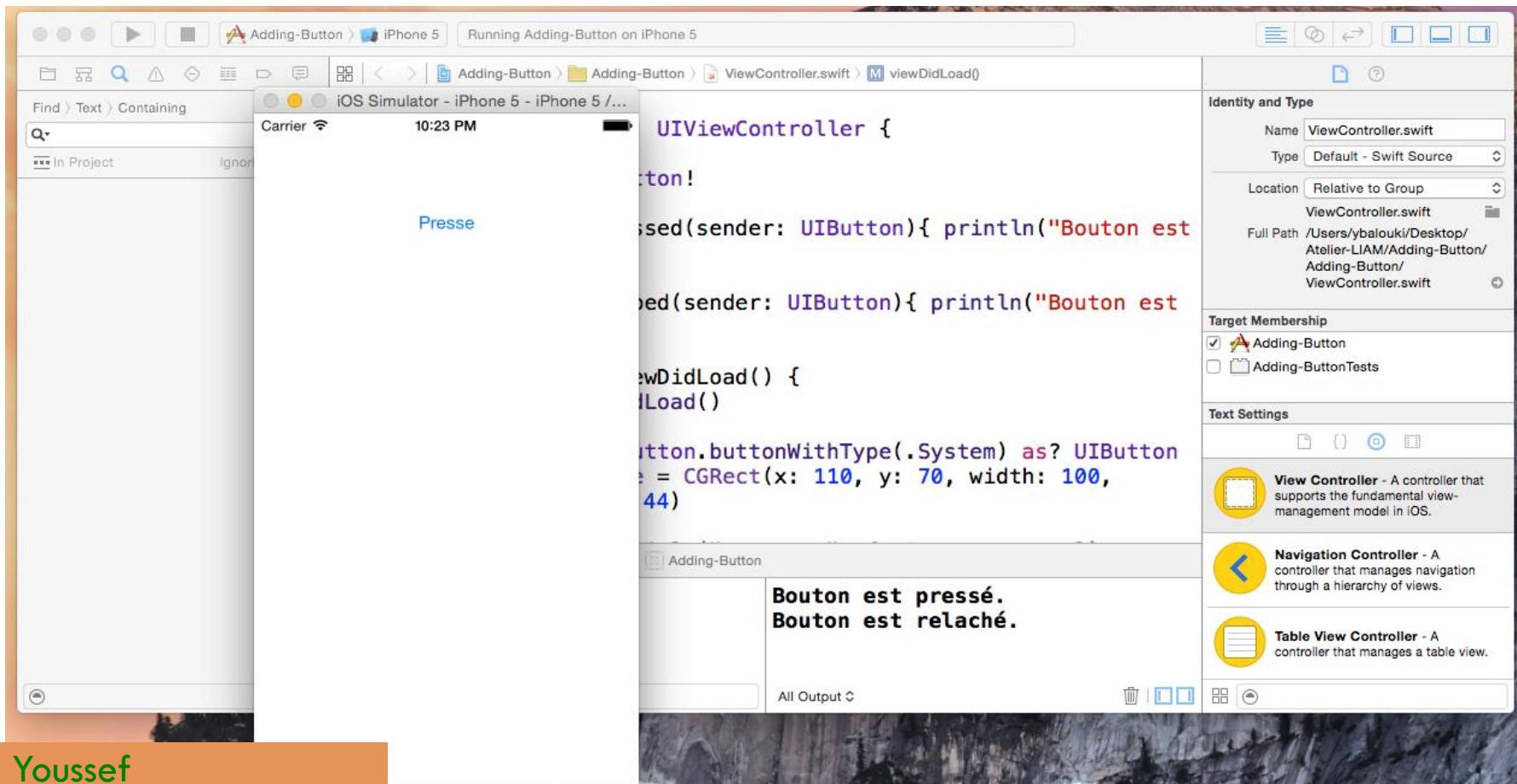
## □ Objectifs :

- ◆ afficher un bouton sur une interface utilisateur
- ◆ gérer les événements tactiles pour ce bouton.

## □ Solution

Utilisation de la classe :**UIButton**.

# Ajout de boutons à l'interface utilisateur avec UIButton



```
Adding-Button > iPhone 5 Running Adding-Button on iPhone 5
iOS Simulator - iPhone 5 - iPhone 5 /...
Carrier: iPhone 5 10:23 PM
Find > Text > Containing: Presse
In Project Ignor
UIViewController {
    @IBOutlet weak var bouton: UIButton!
    @IBAction func boutonTapped(_ sender: UIButton) {
        print("Bouton est pressé")
    }
    @IBAction func boutonReleased(_ sender: UIButton) {
        print("Bouton est relaché")
    }
}
viewDidLoad() {
    super.viewDidLoad()
    self.view.backgroundColor = .white
    self.bouton = UIButton.buttonWithType(.System) as? UIButton
    self.bouton!.frame = CGRectMake(x: 110, y: 70, width: 100, height: 44)
}

Adding-Button
Bouton est pressé.
Bouton est relaché.

All Output
```

The screenshot shows the Xcode interface with a project named "Adding-Button". The main editor displays a Swift file containing code to add a button to a view and handle its tap events. The storyboard shows a single view controller with a button. The output pane shows the console output for the simulator, indicating that the button has been pressed and released. The identity inspector on the right shows the file is named "ViewController.swift" and is a "Default - Swift Source". The target membership section shows the project is targeting "Adding-Button". The text settings section provides links to View Controller, Navigation Controller, and Table View Controller documentation.

## □ Définir le bouton dans le **view controller**

```
import UIKit

class ViewController: UIViewController {

    var button: UIButton!

    func buttonIsPressed(sender: UIButton){
        println("Bouton est pressé.")
    }
    func buttonIsTapped(sender: UIButton){
        println("Bouton est relaché.")
    }
}
```

## □ Définir le bouton dans le **view controller**

```
override func viewDidLoad(){
    super.viewDidLoad()

    button = UIButton.buttonWithType(.System) as? UIButton
    button.frame = CGRect(x: 110, y: 70, width: 100, height: 44)

    // button.setTitle("Presse", forState: .Normal)

    button.setTitle("Presse", forState: UIControlState.Normal)

    button.setTitle("Je suis pressé", forState: .Highlighted)

    button.addTarget(self, action: "buttonIsPressed:", forControlEvents: UIControlEvents.TouchDown)
    //UIControlEvents.TouchDown

    //button.addTarget(self,action: "buttonIsPressed:", forControlEvents: .TouchUpInside)
    button.addTarget(self,action: "buttonIsTapped:", forControlEvents: .TouchUpInside)

    view.addSubview(button)
}

}
```

iOS Simulator - iPhone 5 - iPhone 5 / iOS 8.3 (12F69)

Carrier

10:30 PM



Normale

```
/* 3 */
import UIKit

class ViewController: UIViewController {
    var button: UIButton!

    override func viewDidLoad() {
        super.viewDidLoad()

        let normalImage = UIImage(named: "NormalBlueButton")
        let highlightedImage = UIImage(named: "HighlightedBlueButton")

        button = UIButton.buttonWithType(.Custom) as? UIButton
        button.frame = CGRect(x: 110, y: 70, width: 100, height: 44)
        button.setTitle("Normale", forState: .Normal)
        button.setTitle("Pressé", forState: .Highlighted)

        button.setBackgroundImage(normalImage, forState: .Normal)
        button.setBackgroundImage(highlightedImage, forState: .Highlighted)

        view.addSubview(button)
    }
}
```

# Personnaliser un UIButton

The screenshot shows the Xcode interface with two main panes: the left pane displays the source code for a UIViewController subclass, and the right pane shows the simulator output.

```
import Foundation
import UIKit

class myButton : UIButton {
    override init(frame: CGRect) {
        super.init(frame: frame)
        layer.cornerRadius = 9
        layer.borderWidth = 5
        layer.borderColor = UIColor.red.cgColor
    }

    required init?(coder aDecoder: NSCoder) {
        fatalError("init(coder:) has not been implemented")
    }
}

class ViewController: UIViewController {
    var button :myButton!
    @IBAction func picker(_ sender: UIDatePicker) {
        print(sender.date)
        var formd = DateFormatter()
        formd.dateFormat = "yyyy-MM-dd"
        lbl.text = formd.string(from: sender.date)
    }

    @IBOutlet weak var lbl: UILabel!
    func ButtonIsPressed(sender : myButton){
        print("bouton est presse ")
    }
    func ButtonIsTapped() {
        print("button est relaché ")
    }

    override func viewDidLoad() { super.viewDidLoad()

        //button = UIButton(type: .system )
        button = myButton(frame: CGRect(x: 140, y:280 ,width : 100 ,height : 40))
        button.setTitle("Presse ", for : .normal)
        // button.backgroundColor = UIColor.blue
        button.setTitle("je Presse ", for: .highlighted)
        button.setTitleColor(UIColor.red, for: .normal)
        button.addTarget(self, action: #selector(ButtonIsPressed), for: UIControlEvent.touchUpInside)

        view.addSubview(button)
    }
}
```

The simulator preview on the left shows a red rounded rectangular button labeled "Presse". Below it is a date picker showing the date "2017-05-02".

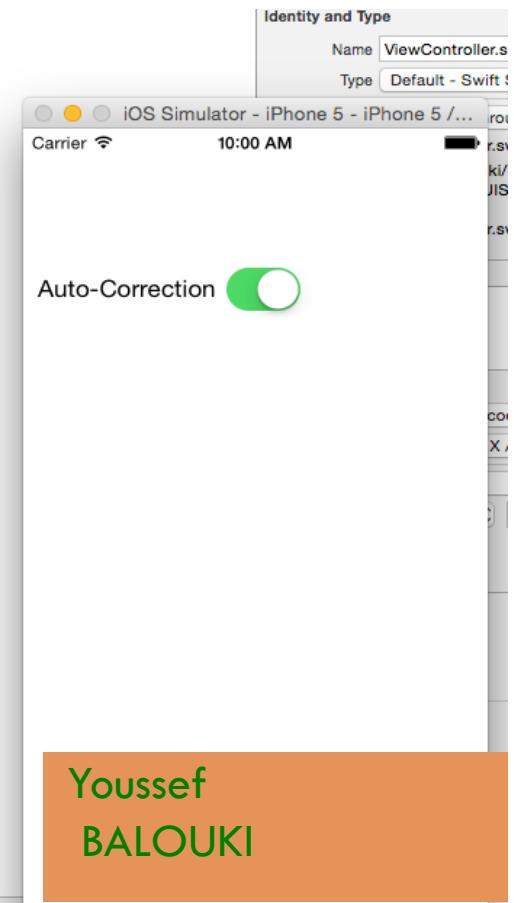
The right pane shows the console output:

```
bouton est presse
2017-05-02 20:54:06 +0000
```

# Création, utilisation et Personnalisation de commutateurs avec : UISwitch

**Objectif : la possibilité d'activer ou désactiver une option.  
: la classe UISwitch**

```
class ViewController: UIViewController {  
  
    var mainSwitch : UISwitch!  
    var lbl : UILabel!  
  
    func switchChanged(sender : UISwitch){  
        println("Sender is = \(sender)")  
        if mainSwitch.on {  
            print("j ai change de valeur On")  
        }  
        else  
        {  
            print("j ai change de valeur Off")  
        }  
    }  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        mainSwitch = UISwitch(frame: CGRect(x: 140, y: 100, width: 100, height: 100))  
        lbl = UILabel(frame: CGRect(x: 10, y: 65, width: 200, height: 100))  
        lbl.text = "Auto-Correction"  
  
        view.addSubview(mainSwitch)  
        view.addSubview(lbl)  
    }  
}
```

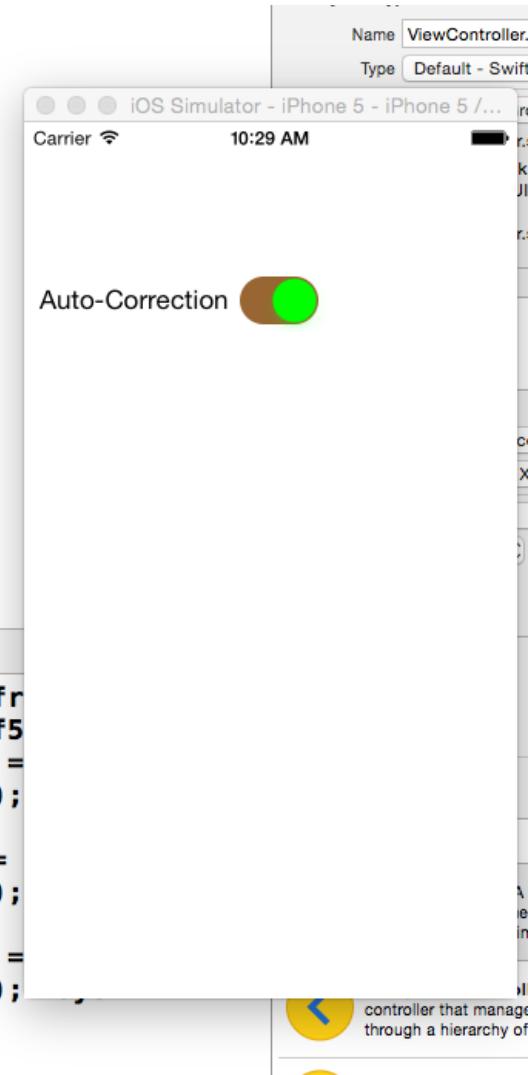


# Création, utilisation et Personnalisation de commutateurs avec : UISwitch

```
lbl = UILabel(frame: CGRect(x: 10, y: 65, width: 200, height: 100))
lbl.text = "Auto-Correction"

view.addSubview(mainSwitch)
view.addSubview(lbl)
```

```
mainSwitch.tintColor = UIColor.redColor()
/* Adjust the on-mode tint color */
mainSwitch.onTintColor = UIColor.brownColor()
/* Also change the knob's tint color */
mainSwitch.thumbTintColor = UIColor.greenColor()
mainSwitch.setOn(true, animated: true)
mainSwitch.addTarget(self, action: "switchChanged:", forControlEventss:
    UIControlEvents.ValueChanged)
```



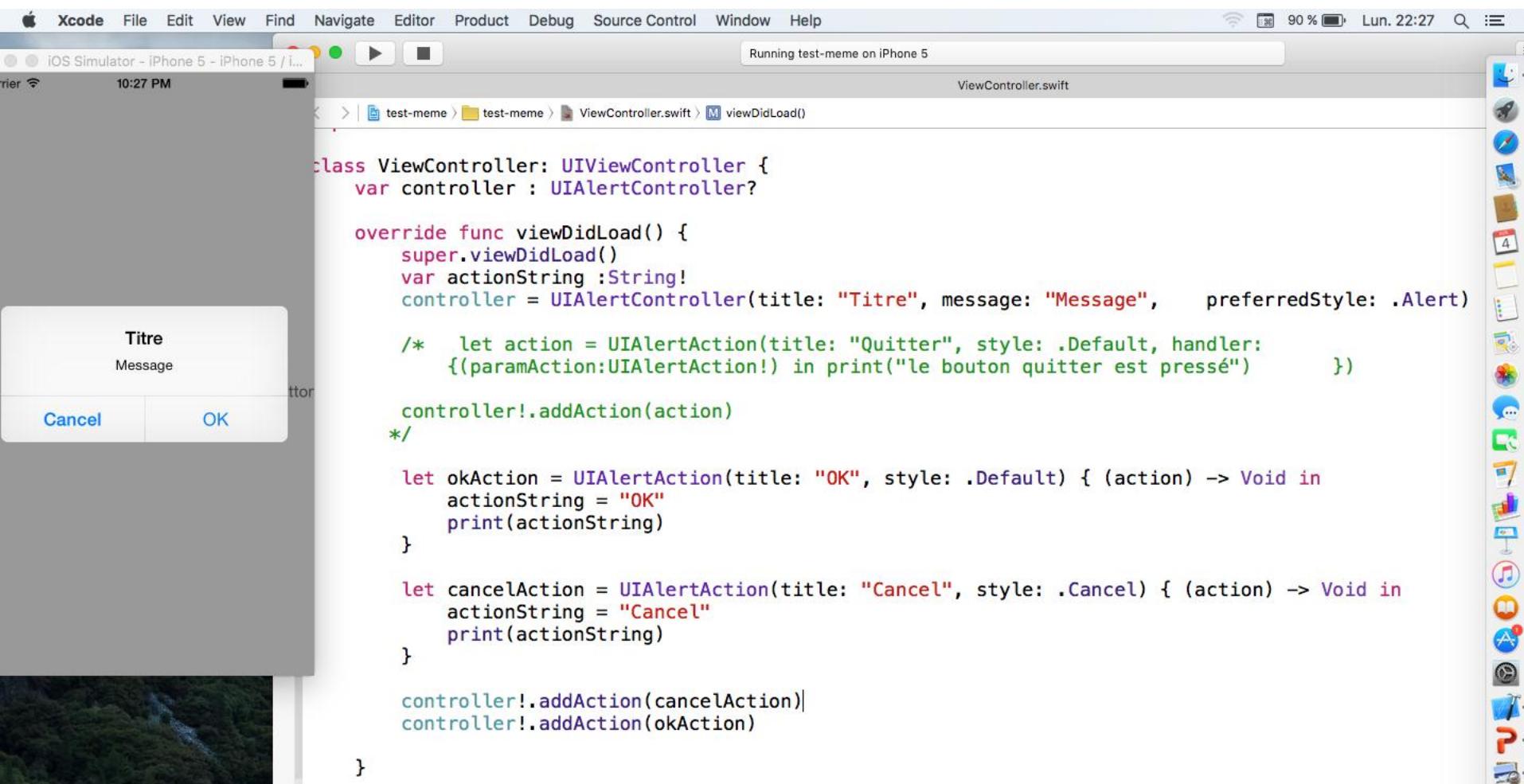
Youssef  
BALOUKI

```
Sender is = <UISwitch: 0x7be5aa30; fr
100; 51 31); layer = <CALayer: 0x7bf5
j ai change de valeur Off?Sender is =
0x7be5aa30; frame = (140 100; 51 31);
<CALayer: 0x7bf5bfa0>>
j ai change de valeur On?Sender is =
0x7be5aa30; frame = (140 100; 51 31);
<CALayer: 0x7bf5bfa0>>
j ai change de valeur Off?Sender is =
0x7be5aa30; frame = (140 100; 51 31);
<CALayer: 0x7bf5bfa0>>
j ai change de valeur On?
```

# Affichage des alerts et des actions (action sheet)

**Objectif** : afficher des vues d'alerte et / ou des actions à l'utilisateur.

## - *UIAlertController*



The screenshot shows the Xcode interface with a Swift file named ViewController.swift open. A modal alert controller is displayed in the foreground, titled "Titre" with the message "Message". It has two buttons: "Cancel" and "OK". The code in the file creates this alert controller and adds actions for both buttons.

```
class ViewController: UIViewController {
    var controller : UIAlertController?

    override func viewDidLoad() {
        super.viewDidLoad()
        var actionString :String!
        controller = UIAlertController(title: "Titre", message: "Message", preferredStyle: .Alert)

        /* let action = UIAlertAction(title: "Quitter", style: .Default, handler:
           {(paramAction:UIAlertAction!) in print("le bouton quitter est pressé") } )

        controller!.addAction(action)
    */

        let okAction = UIAlertAction(title: "OK", style: .Default) { (action) -> Void in
            actionString = "OK"
            print(actionString)
        }

        let cancelAction = UIAlertAction(title: "Cancel", style: .Cancel) { (action) -> Void in
            actionString = "Cancel"
            print(actionString)
        }

        controller!.addAction(cancelAction)
        controller!.addAction(okAction)
    }
}
```

# Affichage des alerts et des actions (action sheet)

les étapes pour créer une vue d'alerte simple ou une action:

- 1. Créez une instance de la classe `UIAlertController` et spécifier si vous voulez une vue d'alerte ou une action.
- 2. Pour chaque action que vous souhaitez ajouter à votre view d'alerte ou à une action (actions sont généralement représentés par un bouton), créer une instance de la classe [`UIAlertAction`](#).
- 3. Ajouter vos actions `UIAlertAction` à votre contrôleur d'alerte en utilisant [`addAction`](#): la méthode de votre contrôleur d'alerte.
- 4. Afficher votre contrôleur d'alerte en utilisant le [`presentViewController: animation: réalisation: méthode`](#)

# Ajouter des champs texts dans Alert View

```
class ViewController: UIViewController {

    var controller: UIAlertController?

    override func viewDidLoad() {
        super.viewDidLoad()

        controller = UIAlertController(title: "Entrer votre nom d'utilisateur", message: "10 caractères au maximum", preferredStyle: .Alert)

        let action = UIAlertAction(title: "Next", style: UIAlertActionStyle.Default, handler: {[weak self] (paramAction:UIAlertAction!) in
            if let textFields = self!.controller?.textFields
            { let theTextFields = textFields as! [UITextField]
                let userName = theTextFields[0].text
                println("Your username is \(userName) \(theTextFields[1].text)")
            }
        })
        controller!.addAction(action)
        controller!.addTextFieldWithConfigurationHandler ( {( textField : UITextField!) in
            textField.placeholder = "XXXXXXXXXX"
        })

        controller!.addTextFieldWithConfigurationHandler ( {( textField1 : UITextField!) in
            textField1.placeholder = "XXXXXXXXXX"
        })
    }

    override func viewDidAppear(animated: Bool) { super.viewDidAppear(animated)
        self.presentViewController(controller!, animated: true, completion: nil)
    }

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



Youssef  
BALOUKI

# Ajouter des actions sheets

```
class ViewController: UIViewController {

    var controller: UIAlertController?

    override func viewDidLoad() {
        super.viewDidLoad()

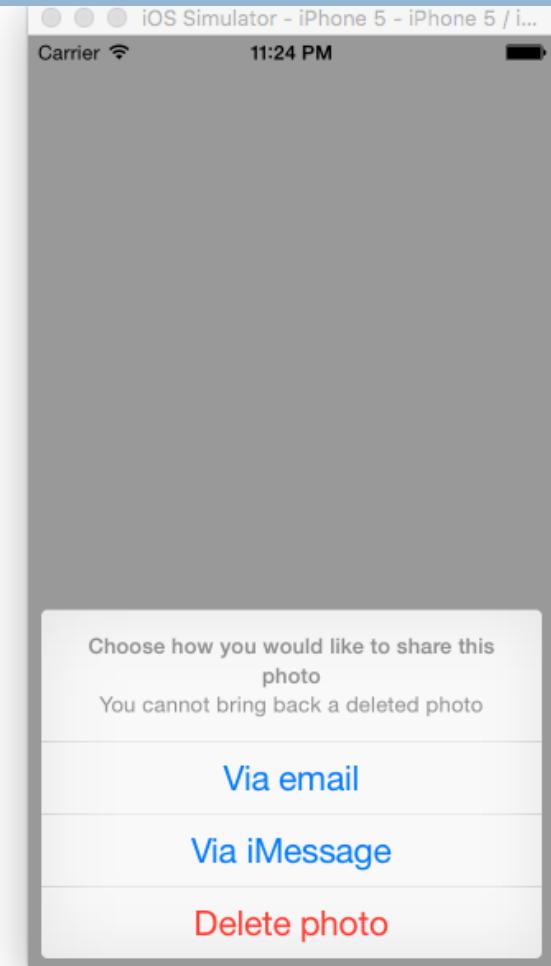
        controller = UIAlertController(
            title: "Choose how you would like to share this photo",
            message: "You cannot bring back a deleted photo",
            preferredStyle: .ActionSheet)

        let actionEmail = UIAlertAction(title: "Via email",
            style: UIAlertActionStyle.Default,
            handler: {(paramAction:UIAlertAction!) in
                /* Send the photo via email */
            })

        let actionIMessage = UIAlertAction(title: "Via iMessage",
            style: UIAlertActionStyle.Default,
            handler: {(paramAction:UIAlertAction!) in
                /* Send the photo via iMessage */
                print("ok photo")
            })

        let actionDelete = UIAlertAction(title: "Delete photo",
            style: UIAlertActionStyle.Destructive,
            handler: {(paramAction:UIAlertAction!) in
                /* Delete the photo here */
            })

        controller!.addAction(actionEmail)
        controller!.addAction(actionIMessage)
        controller!.addAction(actionDelete)
    }
}
```



# Slider

ViewController.swift

Slider

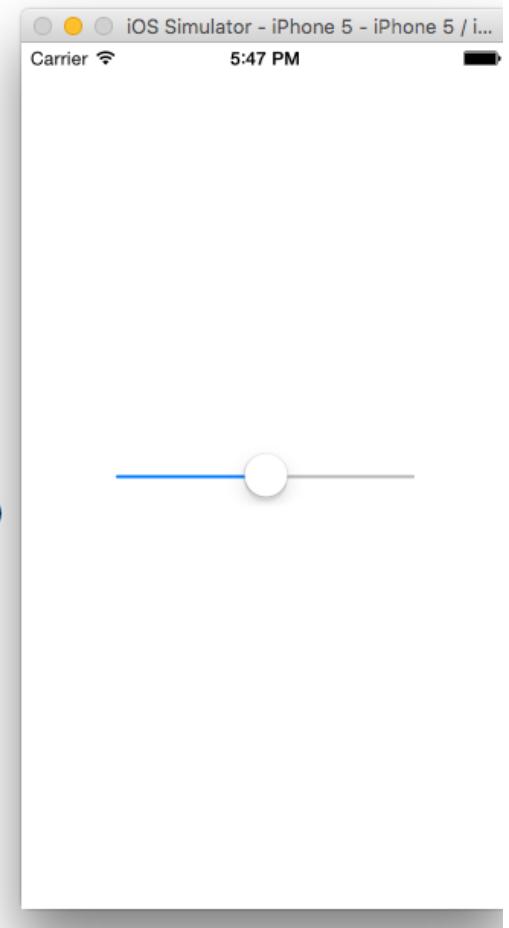
Created by youssef balouki on 12/04/16.  
Copyright (c) 2016 youssef balouki. All rights reserved.

```
import UIKit

class ViewController: UIViewController {

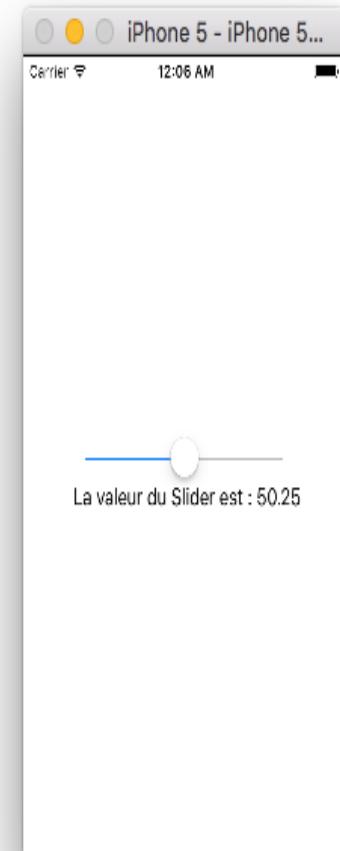
    var slider: UISlider!

    override func viewDidLoad() {
        super.viewDidLoad()
        slider = UISlider(frame: CGRect(x: 0, y: 0, width: 200, height: 23))
        slider.center = view.center
        slider.minimumValue = 0
        slider.maximumValue = 100
        slider.value = slider.maximumValue / 2.0
        view.addSubview(slider)
    }
}
```



# Slider

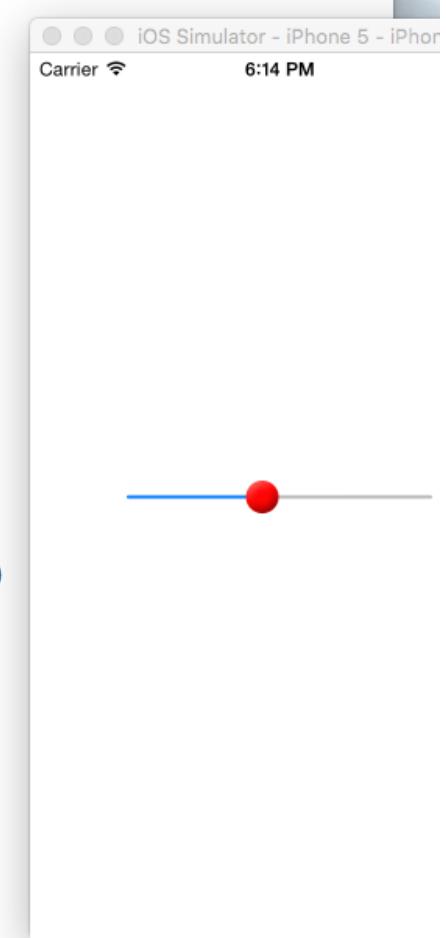
```
class ViewController: UIViewController {
    var slider: UISlider!
    var label : UILabel!
    func sliderValueChanged(slider: UISlider){
        //println("Slider's new value is \(slider.value)")
        label.text = "La valeur du Slider est : \(slider.value)"
    }
    override func viewDidLoad() {
        super.viewDidLoad()
        slider = UISlider(frame: CGRect(x: 0, y: 0, width: 200, height: 23))
        label = UILabel(frame: CGRect(x: 50, y: 300, width: 280, height: 23))
        slider.center = view.center
        slider.minimumValue = 0
        slider.maximumValue = 100
        slider.value = slider.maximumValue / 2.0
        slider.addTarget(self,
                        action: #selector(ViewController.sliderValueChanged(_:)),
                        forControlEvents: .ValueChanged)
        view.addSubview(slider)
        view.addSubview(label)
    }
}
```



# Slider

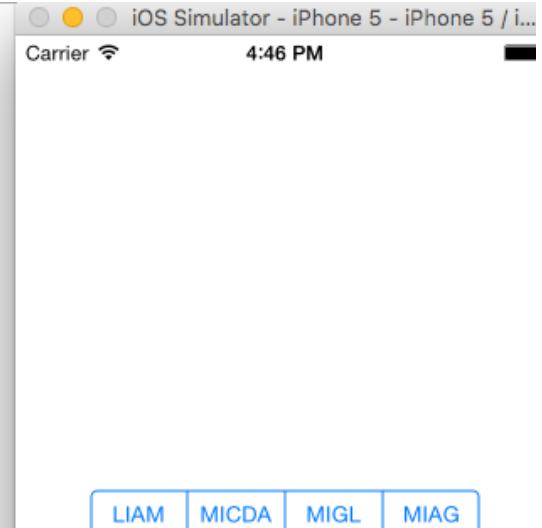
```
action: "sliderValueChanged:", forControlEvents: .ValueChanged)
view.addSubview(slider)
view.addSubview(label)
}
```

```
3 parties
ss ViewController: UIViewController {
    slider: UISlider!
    override func viewDidLoad() {
        super.viewDidLoad()
        slider = UISlider(frame: CGRect(x: 0, y: 0, width: 200, height: 23))
        slider.center = view.center
        slider.minimumValue = 0
        slider.maximumValue = 100
        slider.value = slider!.maximumValue / 2.0
        slider.setThumbImage(UIImage(named: "Image"), forState: .Normal)
        slider.setThumbImage(UIImage(named: "Image-1"), forState: .Highlighted)
        view.addSubview(slider)
    }
}
```



# Regroupement d'options avec UISegmentedControl

```
//  
// ViewController.swift  
// segmentedControl  
//  
// Created by youssef balouki on 13/04/16.  
// Copyright (c) 2016 youssef balouki. All rights reserved.  
  
import UIKit  
  
class ViewController: UIViewController {  
  
    var segmentedControl: UISegmentedControl!  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        let segments = [ "LIAM", "MICDA", "MIGL", "MIAG"]  
        segmentedControl = UISegmentedControl(items: segments)  
        segmentedControl.center = view.center  
        self.view.addSubview(segmentedControl)  
    }  
  
}
```



# Regroupement d'options avec UISegmentedControl

```
import UIKit

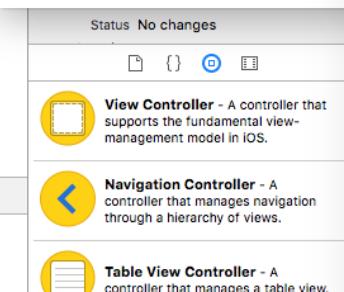
class ViewController: UIViewController {

    var segmentedControl: UISegmentedControl!

    func segmentedControlValueChanged(sender: UISegmentedControl) {
        let selectedSegmentIndex = sender.selectedSegmentIndex
        let selectedSegmentText = sender.titleForSegmentAtIndex(selectedSegmentIndex)
        println("Segment \(selectedSegmentIndex) avec le texte " + "\(selectedSegmentText!) est sélectionné")
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        let segments = ["LIAM", "MICDA", "MIGL", "MIAG"]
        segmentedControl = UISegmentedControl(items: segments)
        segmentedControl.center = view.center
        segmentedControl.addTarget(self, action: "segmentedControlValueChanged:", forControlEvents: UIControlEvents.ValueChanged)
        self.view.addSubview(segmentedControl)
    }
}
```

LIAM MICDA MIGL MIAG



# Regroupement d'options avec UISegmentedControl

```
UIControlEvents.ValueChanged)
    self.view.addSubview(segmentedControl)

}

}

import UIKit

class ViewController: UIViewController {

    var segmentedControl: UISegmentedControl!

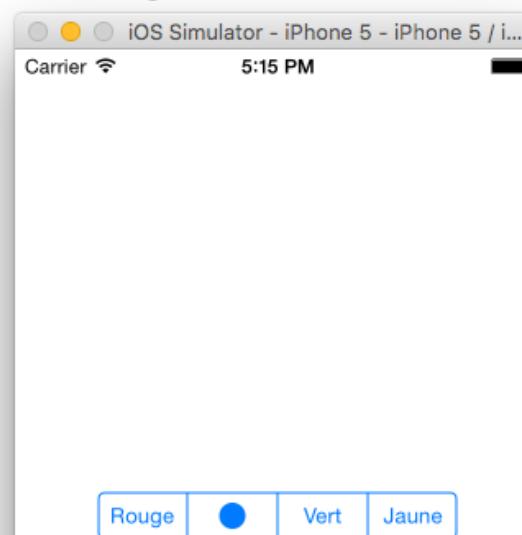
    override func viewDidLoad() {
        super.viewDidLoad()

        let segments = ["Rouge", UIImage(named: "Image")!, "Vert", "Jaune"]

        segmentedControl = UISegmentedControl(items: segments )
        segmentedControl.center = view.center
        self.view.addSubview(segmentedControl)

    }

}
```



# les options de partage avec UIActivityViewController

```
import UIKit
class ViewController: UIViewController, UITextFieldDelegate {
    var textField:UITextField!
    var buttonShare:UIButton!
    var activityViewController:UIActivityViewController!

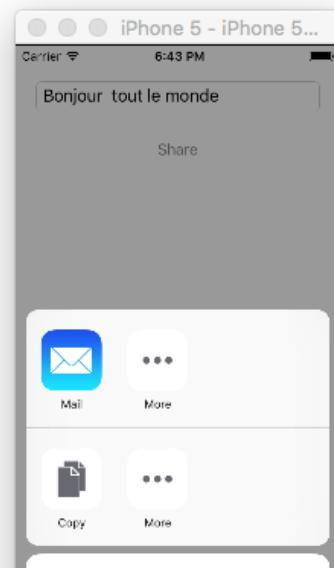
    func handleShare(sender: UIButton){
        if (textField.text!.isEmpty) {

            let message = "Please enter a text and then press Share"
            let alertController = UIAlertController(title: nil, message: message, preferredStyle: .Alert)
            alertController.addAction(UIAlertAction(title: "OK", style: .Default, handler: nil))
            presentViewController(alertController, animated: true, completion: nil)
            return
        }

        activityViewController = UIActivityViewController(activityItems: [textField.text! as NSString],
                                                        applicationActivities: nil)
        presentViewController(activityViewController, animated: true, completion: { })
    }

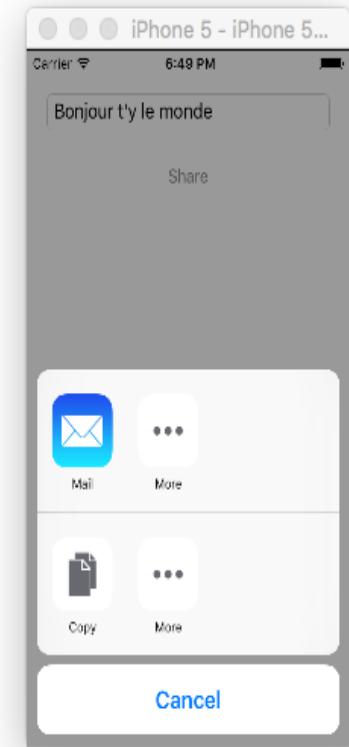
    func createTextField(){
        textField = UITextField(frame:CGRect(x: 20, y: 35, width: 280, height: 30))
        textField.borderStyle = .RoundedRect
        textField.placeholder = "Enter text to share..."
        textField.delegate = self
        view.addSubview(textField)
    }

    func createButton(){
        buttonShare = UIButton.init(type: UIButtonType.System)
        //buttonWithType(.System)
        buttonShare.frame = CGRect(x: 20, y: 80, width: 280, height: 44)
        buttonShare.setTitle("Share", forState:.Normal)
        buttonShare.addTarget(self, action: #selector(ViewController.handleShare(_:)),
                            forControlEvents:.TouchUpInside)
        view.addSubview(buttonShare)
    }
}
```



# les options de partage avec UIActivityViewController

```
func createTextField(){  
    textField = UITextField(frame:CGRect(x: 20, y: 35, width: 280, height: 30))  
    textField.borderStyle = .RoundedRect  
    textField.placeholder = "Enter text to share..."  
    textField.delegate = self  
    view.addSubview(textField)    }  
  
func createButton(){  
    buttonShare = UIButton.init(type: UIButtonType.System)  
    // buttonWithType(.System)  
    buttonShare.frame = CGRect(x: 20, y: 80, width: 280, height: 44)  
    buttonShare.setTitle("Share", forState:.Normal)  
    buttonShare.addTarget(self, action: #selector(ViewController.handleShare(_:)),  
                         forControlEvents:.TouchUpInside)  
    view.addSubview(buttonShare)    }  
  
override func viewDidLoad() {  
    super.viewDidLoad()  
    createTextField()  
    createButton()    }  
  
func textFieldShouldReturn(textField: UITextField) -> Bool{  
    textField.resignFirstResponder()  
    return true    }  
}
```



# Ajout d'une image au navigation Bar

**Objectif :** Vous souhaitez afficher une image au lieu de texte comme titre du contrôleur de vue en cours sur le contrôleur de navigation.

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

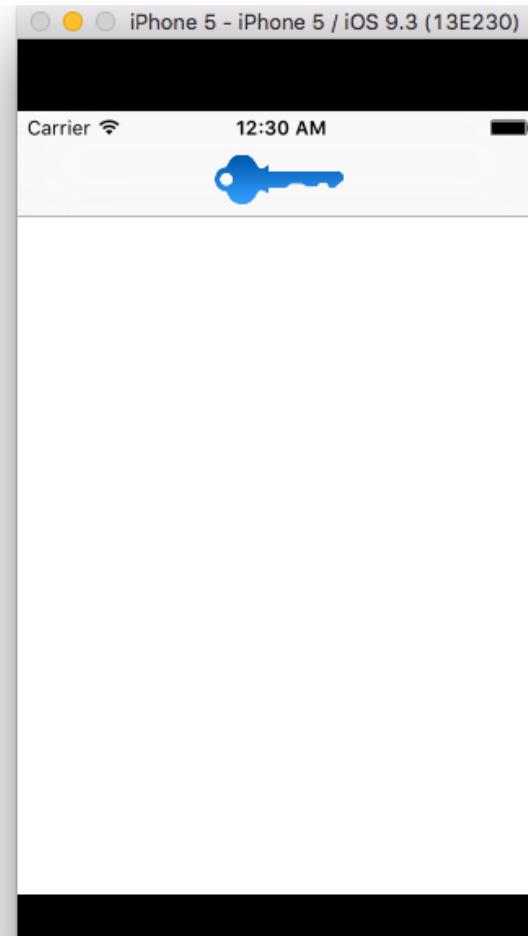
        /* Create an Image View to replace the Title View */
        let imageView = UIImageView(
            frame: CGRect(x: 0, y: 0, width: 100, height: 40))

        imageView.contentMode = .ScaleAspectFit

        let image = UIImage(named:"Logo")

        imageView.image = image

        /* Set the Title View */
        navigationItem.titleView = imageView
    }
}
```



# Ajout du bouton au navigation Bar avec UIBarButtonItem

```
/* 1 */
import UIKit

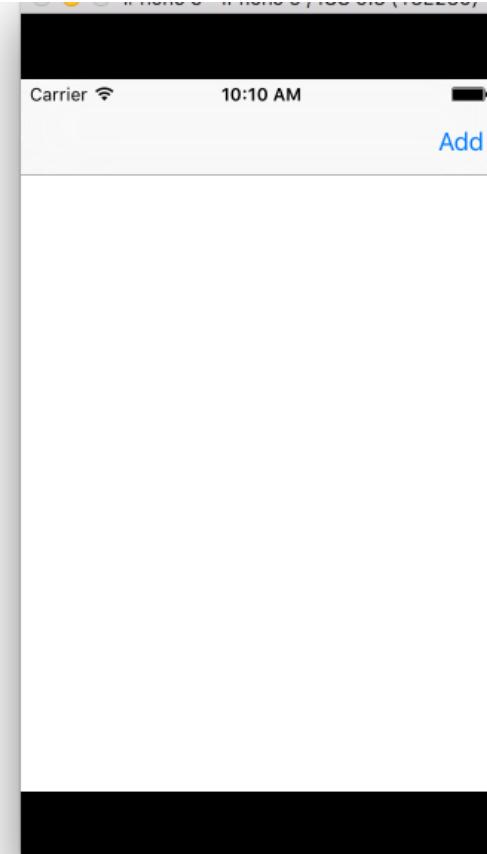
class ViewController: UIViewController {

    func performAdd(sender: UIBarButtonItem){
        print("la méthode ajout est applée")
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        navigationItem.rightBarButtonItem = UIBarButtonItem(
            title: "Add", style: .Plain, target: self,
            action: #selector(ViewController.performAdd(_:)))
    }
}

// /* 2 */
// import UIKit
// 
// class ViewController: UIViewController {
// 
//     func performAdd(sender: UIBarButtonItem){
//         print("Add method got called")
//     }
// }
```



□ ▶ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ Adding Buttons to Navigation Bars Using UIBarButtonItem

# Ajout du bouton + au navigation Bar avec UIBarButtonItem

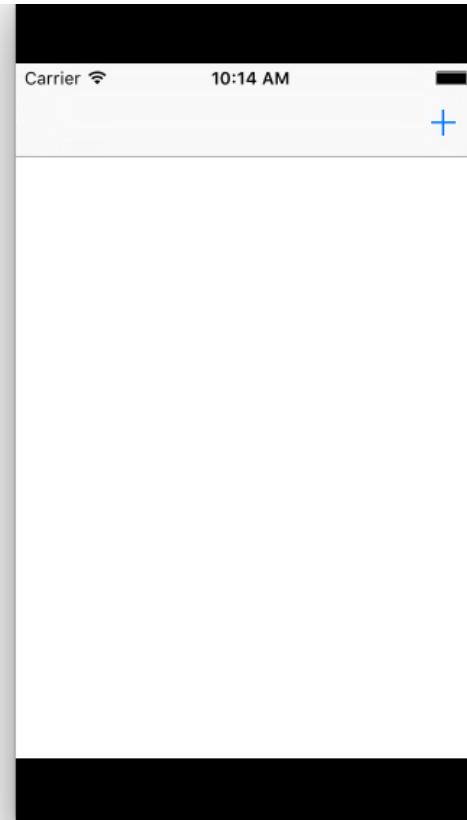
```
import UIKit

class ViewController: UIViewController {

    func performAdd(sender: UIBarButtonItem){
        print("la méthode ajout est applée")
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        navigationItem.rightBarButtonItem = UIBarButtonItem(
            barButtonSystemItem: .Add,
            target: self,
            action: #selector(ViewController.performAdd(_:)))
    }
}
```



Adding Buttons to Navigation Bars Using UIBarButtonItem

la méthode ajout est applée  
la méthode ajout est applée

# Ajout switch au navigation Bar avec UIBarButtonItem

```
class ViewController: UIViewController {

    func switchIsChanged(sender: UISwitch){
        if sender.on{
            print("Switch is on")
        } else {
            print("Switch is off")
        }
    }

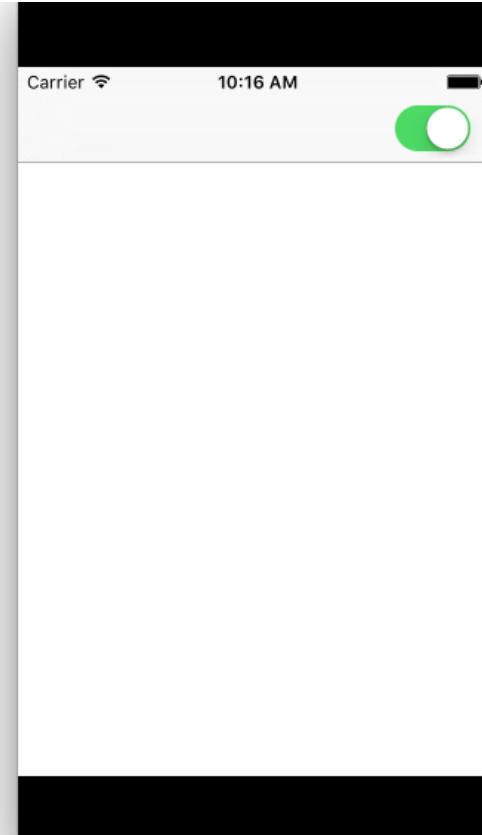
    override func viewDidLoad() {
        super.viewDidLoad()

        let simpleSwitch = UISwitch()
        simpleSwitch.on = true

        simpleSwitch.addTarget(self,
            action: #selector(ViewController.switchIsChanged(_:)),
            forControlEvents: .ValueChanged)

        self.navigationItem.rightBarButtonItem =
            UIBarButtonItem(customView: simpleSwitch)
    }
}

///* 4 */
```



Switch is off

# Ajout du segment au navigation Bar avec UIBarButtonItem

```
class ViewController: UIViewController {

    let items = ["Up", "Down"]

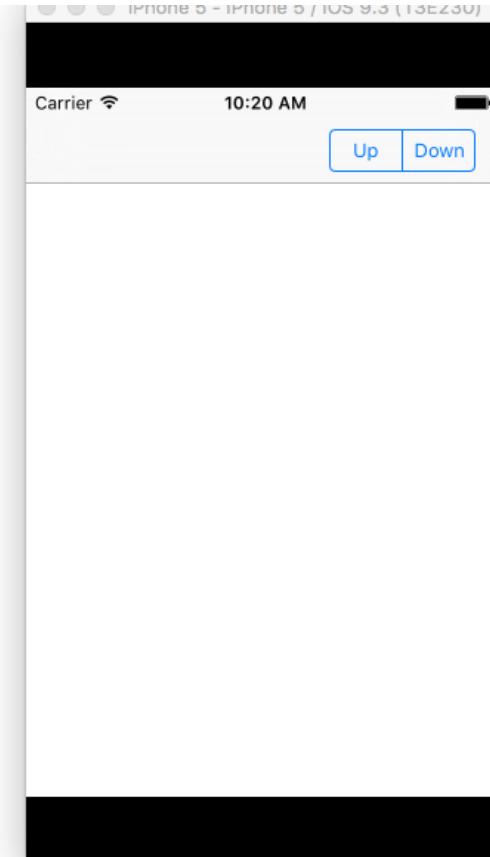
    func segmentedControlTapped(sender: UISegmentedControl) {
        if sender.selectedSegmentIndex < items.count {
            print(items[sender.selectedSegmentIndex])
        } else {
            print("Unknown button is pressed")
        }
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        let segmentedControl = UISegmentedControl(items: items)
        segmentedControl.momentary = true

        segmentedControl.addTarget(self,
            action: #selector(ViewController.segmentedControlTapped(_:)),
            forControlEvents: .ValueChanged)

        navigationItem.rightBarButtonItem =
            UIBarButtonItem(customView: segmentedControl)
    }
}
```



# Ajout du segment au navigation Bar avec UIBarButtonItem

```
/   segmentedControl.momentary = true
/
/   segmentedControl.addTarget(self,
/     action: #selector(ViewController.segmentedControlTapped(_:)),
/     forControlEvents: .ValueChanged)
/
/   navigationItem.rightBarButtonItem =
/     UIBarButtonItem(customView: segmentedControl)
/
/ }
/
//5
import UIKit

class ViewController: UIViewController {

let items = ["Up", "Down"]

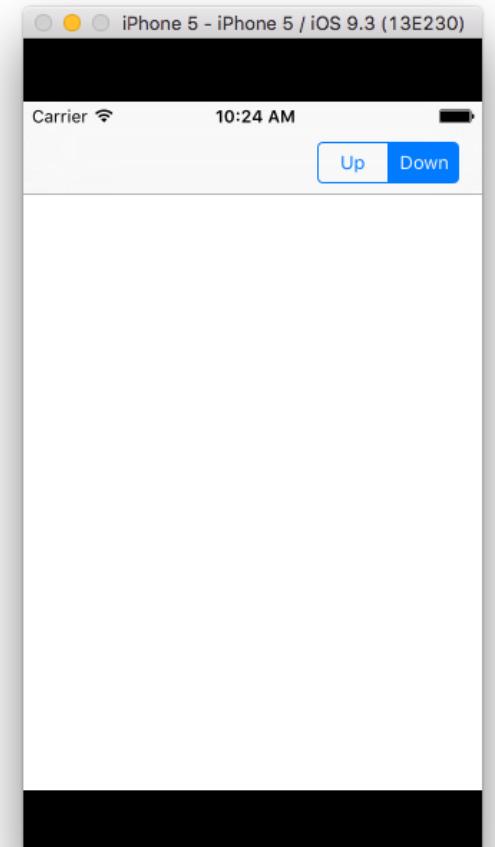
override func viewDidAppear(animated: Bool){
super.viewDidAppear(animated)

let segmentedControl = UISegmentedControl(items: items)

let rightBarButtonItem =
UIBarButtonItem(customView:segmentedControl)

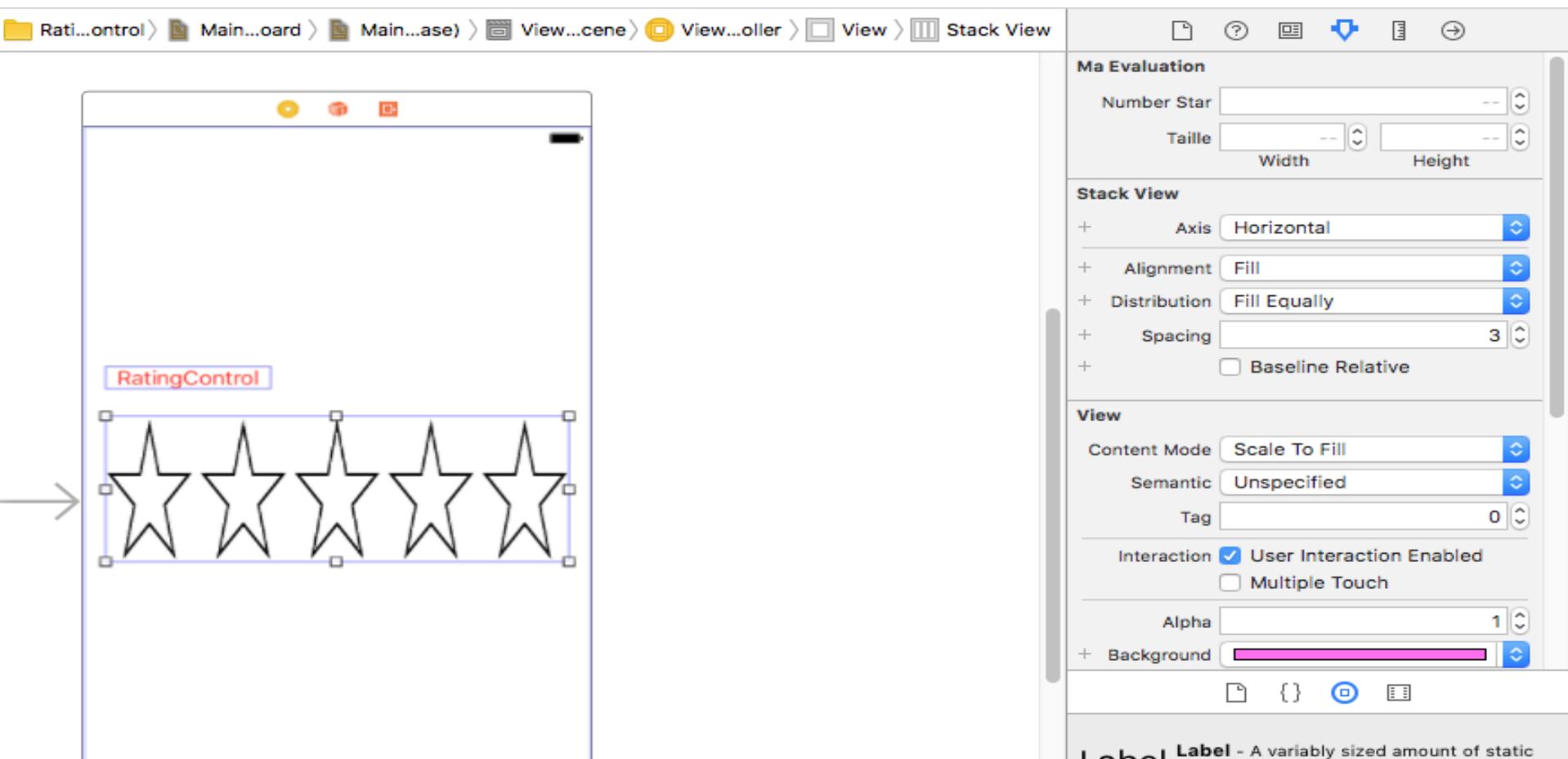
navigationItem.setRightBarButtonItem(rightBarButtonItem, animated: true)

}
```



# Rating Contrôle

Step 1: Création d'un Horizontal Stack view



# Rating Contrôle

## Step 2 : Crédation d'une classe de type StackView

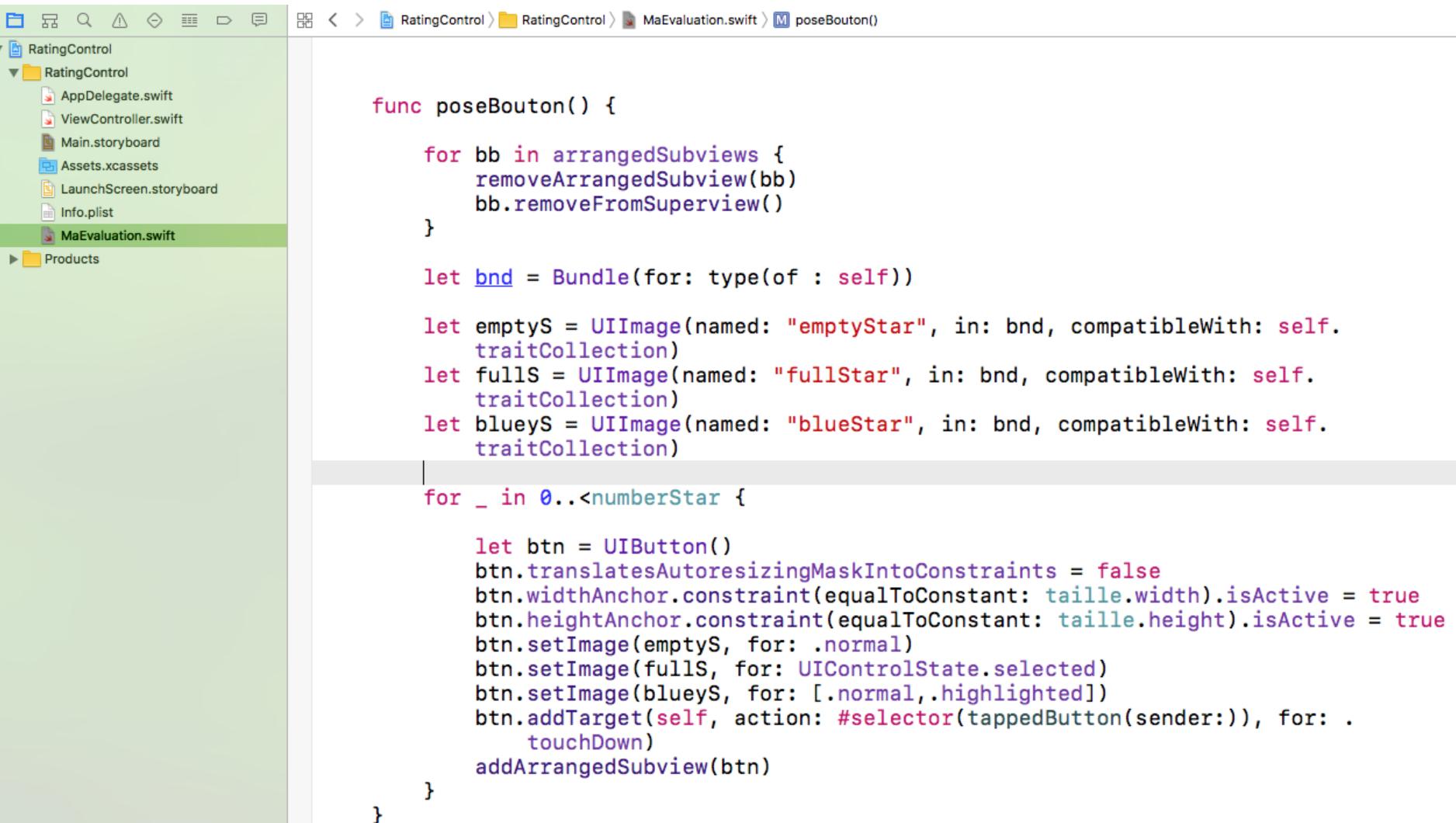
```
RatingControl
RatingControl
  AppDelegate.swift
  ViewController.swift
  Main.storyboard
  Assets.xcassets
  LaunchScreen.storyboard
  Info.plist
MaEvaluation.swift
Products

import UIKit

// annotation IBDesignable permet l'affichage de la classe MaEvaluation
// sur le storyBorad
@IBDesignable class MaEvaluation: UIStackView {
    // annotation IBInspectable permet d'afficher l'attribut numberStar dans l'inspecteur
    @IBInspectable var numberStar : Int = 5 {
        didSet {
            poseBouton()
        }
    }
    @IBInspectable var taille : CGSize = CGSize(width: 44.0, height: 44.0) {
        didSet {
            poseBouton()
        }
    }
    var evalution = 0 {
        didSet {
            remplir()
        }
    }
    override init(frame: CGRect) {
        super.init(frame: frame)
        poseBouton()
    }
    required init(coder: NSCoder) {
        super.init(coder: coder)
        poseBouton()
    }
}
```

# Rating Contrôle

## Step 3 : Définir une fonction qui pose des boutons étoiles dans le StackView



The screenshot shows the Xcode interface with the project structure on the left and the code editor on the right.

**Project Structure:**

- RatingControl (selected)
- RatingControl (grouped)
- AppDelegate.swift
- ViewController.swift
- Main.storyboard
- Assets.xcassets
- LaunchScreen.storyboard
- Info.plist
- MaEvaluation.swift

**Code Editor (MaEvaluation.swift):**

```
func poseBouton() {

    for bb in arrangedSubviews {
        removeArrangedSubview(bb)
        bb.removeFromSuperview()
    }

    let bnd = Bundle(for: type(of: self))

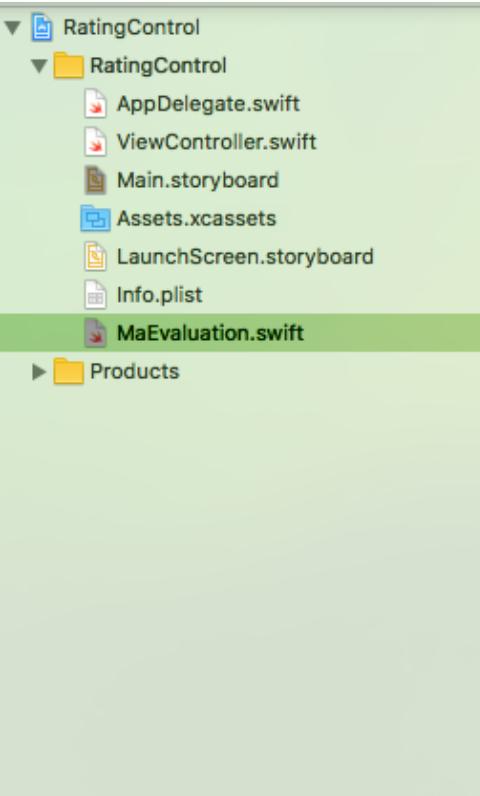
    let emptyS = UIImage(named: "emptyStar", in: bnd, compatibleWith: self.
        traitCollection)
    let fullS = UIImage(named: "fullStar", in: bnd, compatibleWith: self.
        traitCollection)
    let blueyS = UIImage(named: "blueStar", in: bnd, compatibleWith: self.
        traitCollection)

    for _ in 0..<numberStar {

        let btn = UIButton()
        btn.translatesAutoresizingMaskIntoConstraints = false
        btn.widthAnchor.constraint(equalToConstant: taille.width).isActive = true
        btn.heightAnchor.constraint(equalToConstant: taille.height).isActive = true
        btn.setImage(emptyS, for: .normal)
        btn.setImage(fullS, for: UIControlState.selected)
        btn.setImage(blueyS, for: [.normal,.highlighted])
        btn.addTarget(self, action: #selector(tappedButton(sender:)), for: .
            touchDown)
        addArrangedSubview(btn)
    }
}
```

# Rating Contrôle

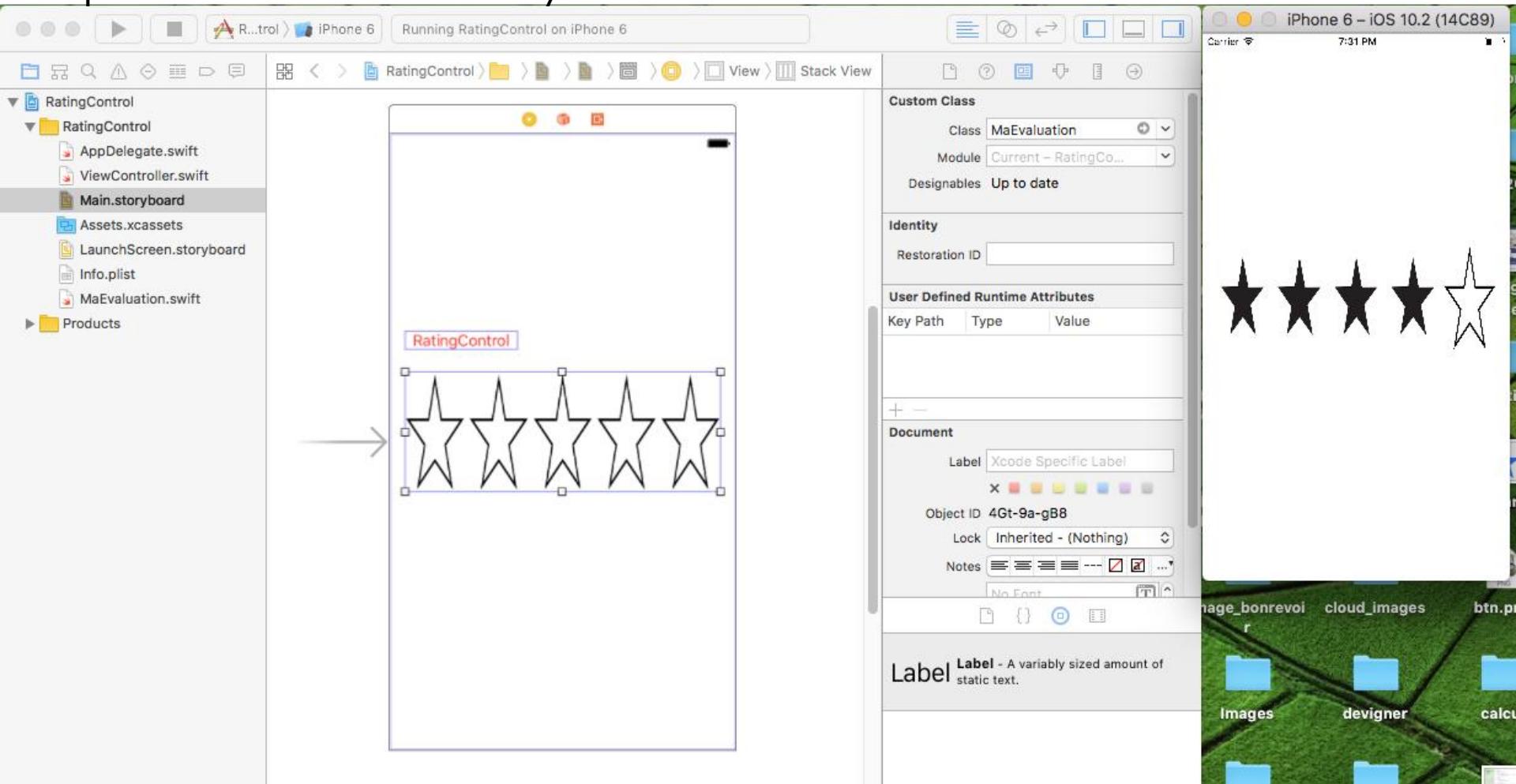
Step4 : Définir l'action de vote (choisir le nombre des étoiles)



```
func tappedButton(sender : UIButton){  
    let current = (arrangedSubviews.index(of: sender))  
    if current == evalution {  
        evalution = -1  
    } else { evalution = current!}  
  
    remplir()  
}  
  
func remplir() {  
    for i in 0..<numberStar {  
        let btn = arrangedSubviews[i] as! UIButton  
        btn.isSelected = i <= evalution  
    }  
}
```

# Rating Contrôle

Step 6 : Lier le StackView sur storyBoard avec la classe MaEvaluation



# Stockage persistants : CoreData

- **Core Data est une base de données locale, munie d'une API orientée objet.**
  - les données sont stockées dans une base de données **SQLite**.
  - les données sont stockées sur le téléphone et pas sur un serveur distant, cloud ou ....
  - Core Data transforme tout en objets Swift, puis les stocke en SQLite.

# Core Data

- Complex/Offline
- Enregistrer et Annuler(undo)
- Tri et Filtrage
- Réduction du code
- Gestion optimale de la mémoire
- Affichage facile des données
- Data GUI (prend en charge les types de stockage persistants :XML & SQLite)

# Core Data

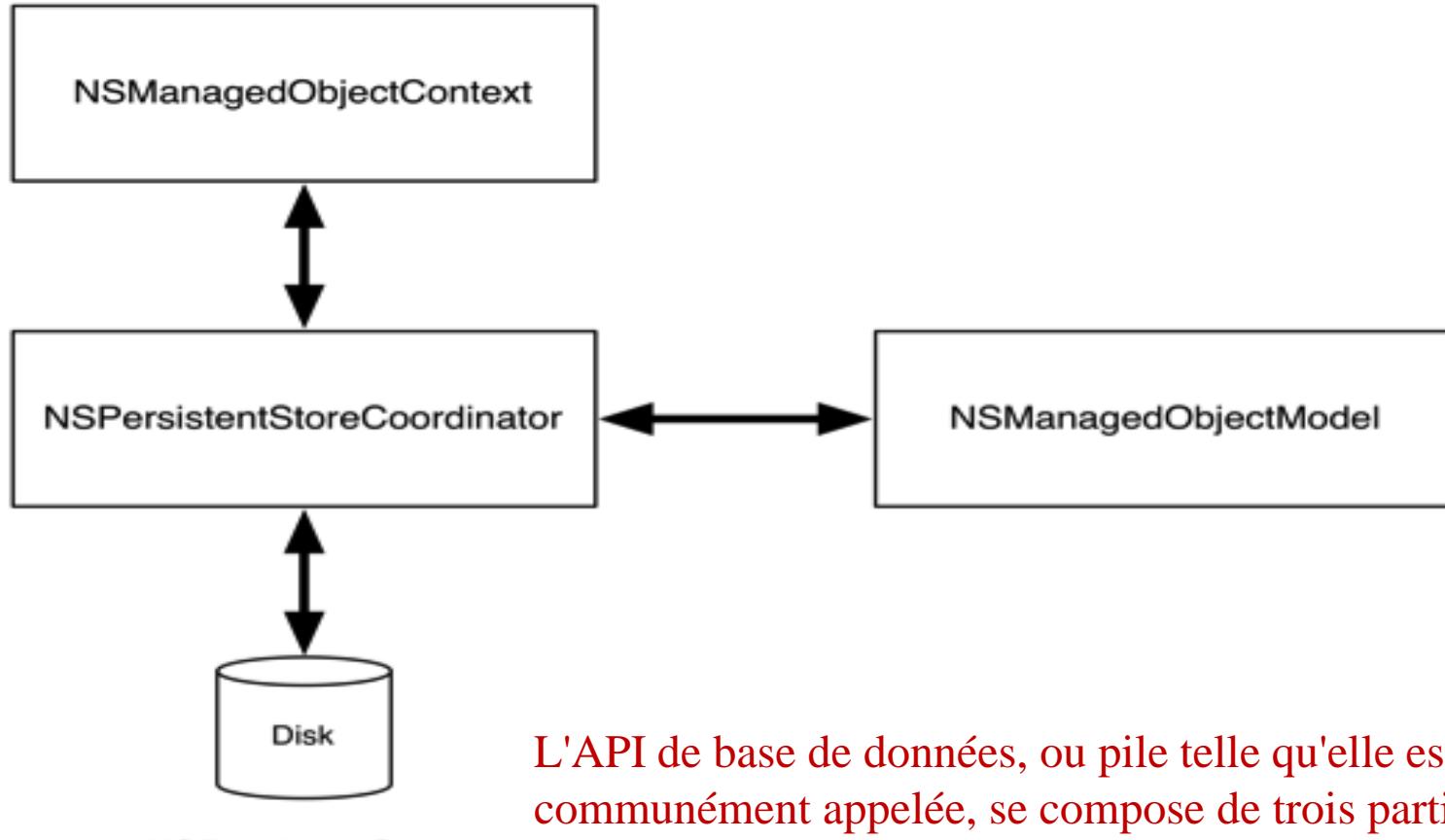
Core Data est considéré comme :

- Un graph d'objets conçu pour gérer les objets de données.
- Un cadre de mapping objet-relationnel

Core Data a deux fonctions principal :

- gérer le graphe d'objets.
- La persistance des données sur le disque est une fonction secondaire, bien que vitale.

# La structure d' API Core Data



L'API de base de données, ou pile telle qu'elle est communément appelée, se compose de trois parties principales: **NSManagedObjectModel**, **NSPersistentStoreCoordinator** et **NSManagedObjectContext**,

# Core Data

- Classes :
  - NSManagedObjectModel
  - NSPersistentStoreCoordinator
  - NSManagedObjectContext
  - NSManagedObject
  - NSPersistentContainer
- Fichiers
  - DataModel.xcdatamodeld
  - DataModel.momd
  - DataModel.sqlite

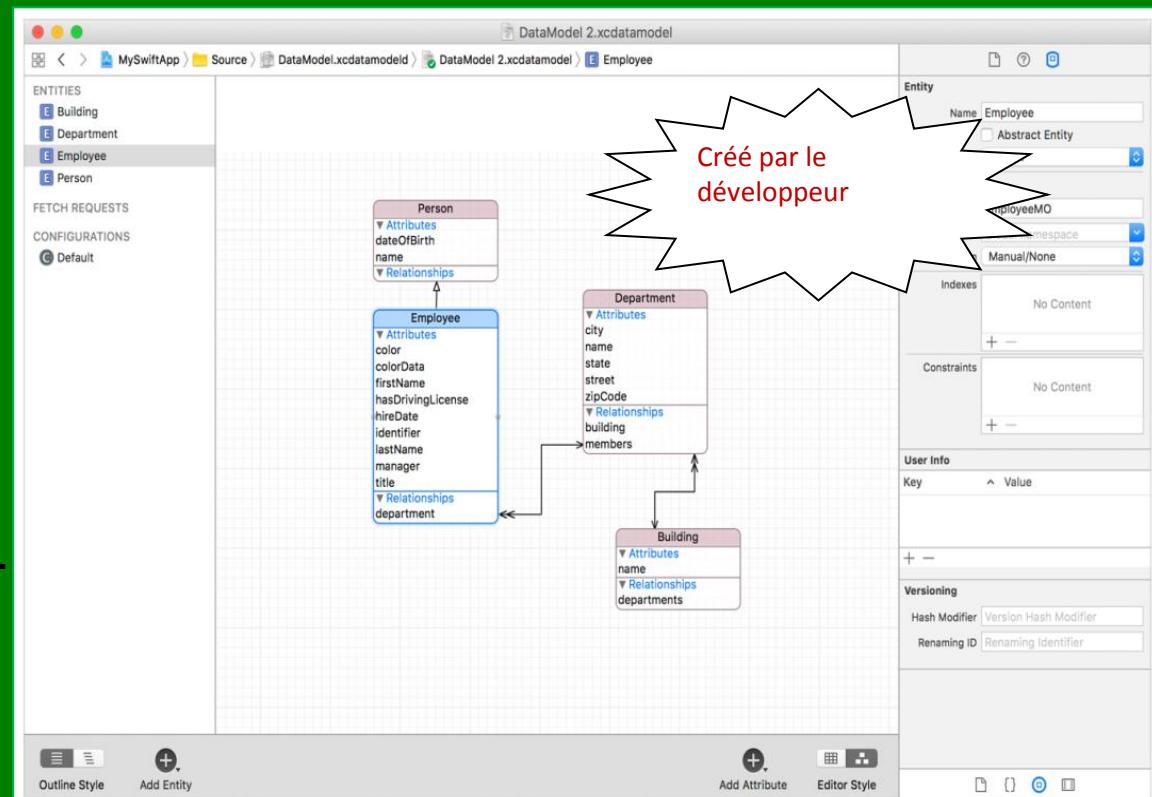
# NSManaged ModelObject

S'initialiser avec



ModelDonnees.momd

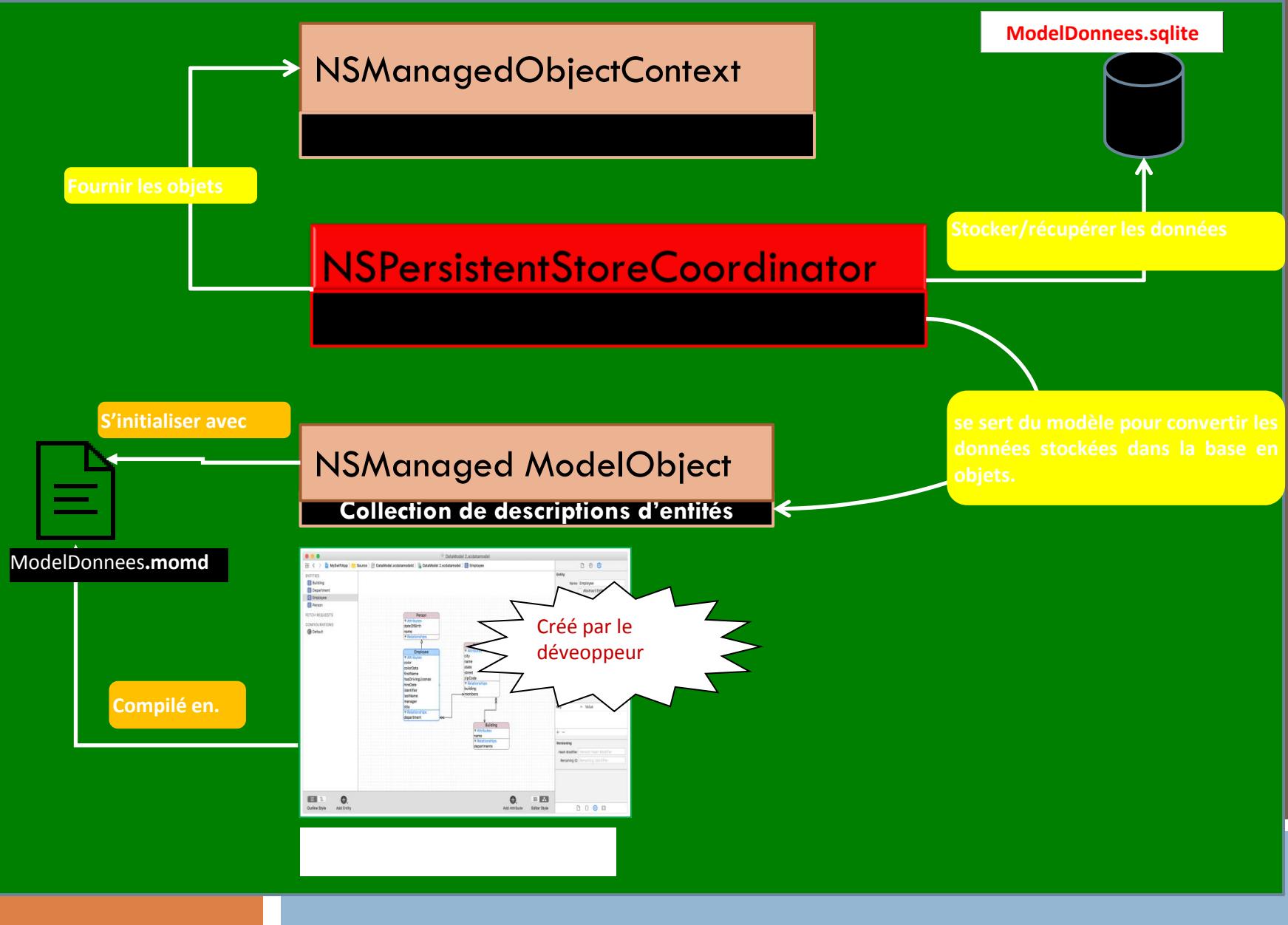
Compilé en.

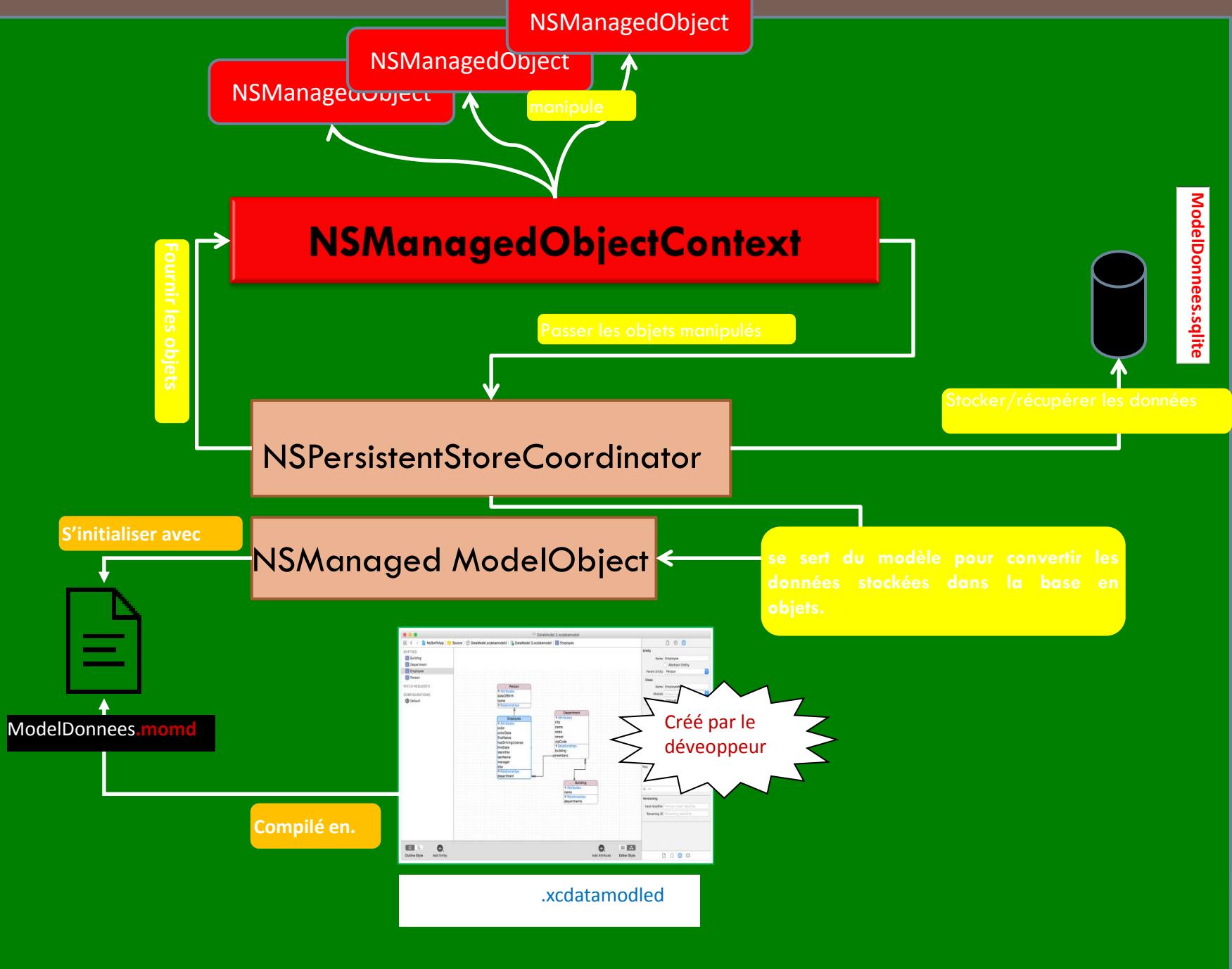


.xcdatamodeld

# Stockage persistants : CoreData

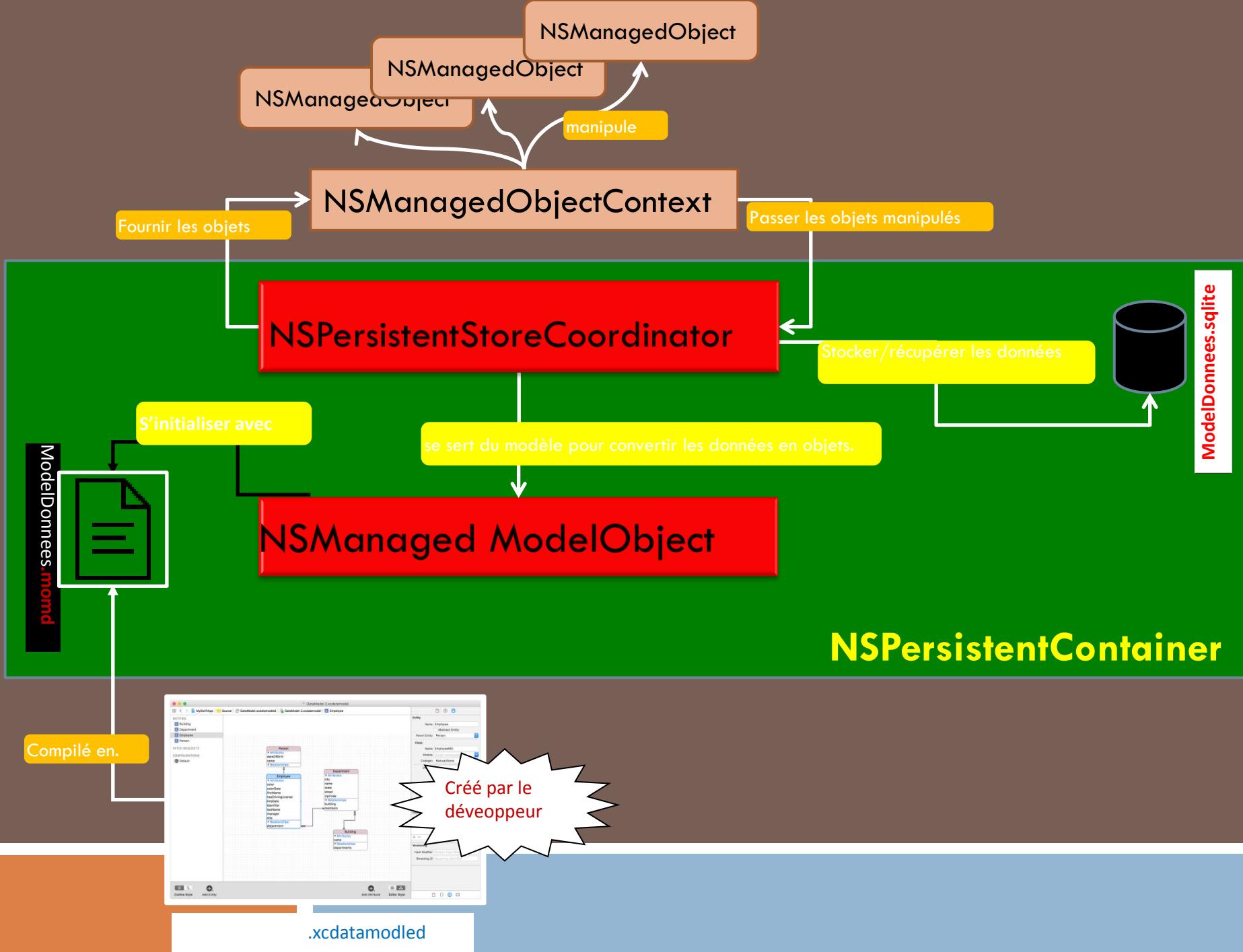
- Une instance de **NSManagedObjectContext** représente un seul "**espace objet**" dans une application. Sa principale responsabilité est de gérer une collection d'objets. Ces objets forment un groupe d'objets de modèle apparentés qui représentent une vue cohérente en interne d'un ou plusieurs stockage persistants. Une seule instance d'objet (**géré**) existe dans un seul et même contexte, mais plusieurs copies d'un objet peuvent exister dans différents contextes. Ainsi, l'élimination des objets est portée dans un contexte particulier.





# NPersistentCoordinator-iOS10

- 1) Créer un fichier DataModel.xcdatamodeld.
- 2) Récupérer la version compilée **DataModel.momd**.
- 3) Initialiser une instance de **NSManagedModel** à partir de cette version compilée.
- 4) Créer un fichier pour notre base de données **DataModel.sqlite**
- 5) Initialiser une instance de **NPersistentStoreCoordinator** à partir de notre fichier **DataModel.sqlite** et de notre instance de **NSManagedModel**.
- 6) Récupérer un contexte de notre **NPersistentStoreCoordinator** avec lequel on va pouvoir travailler.

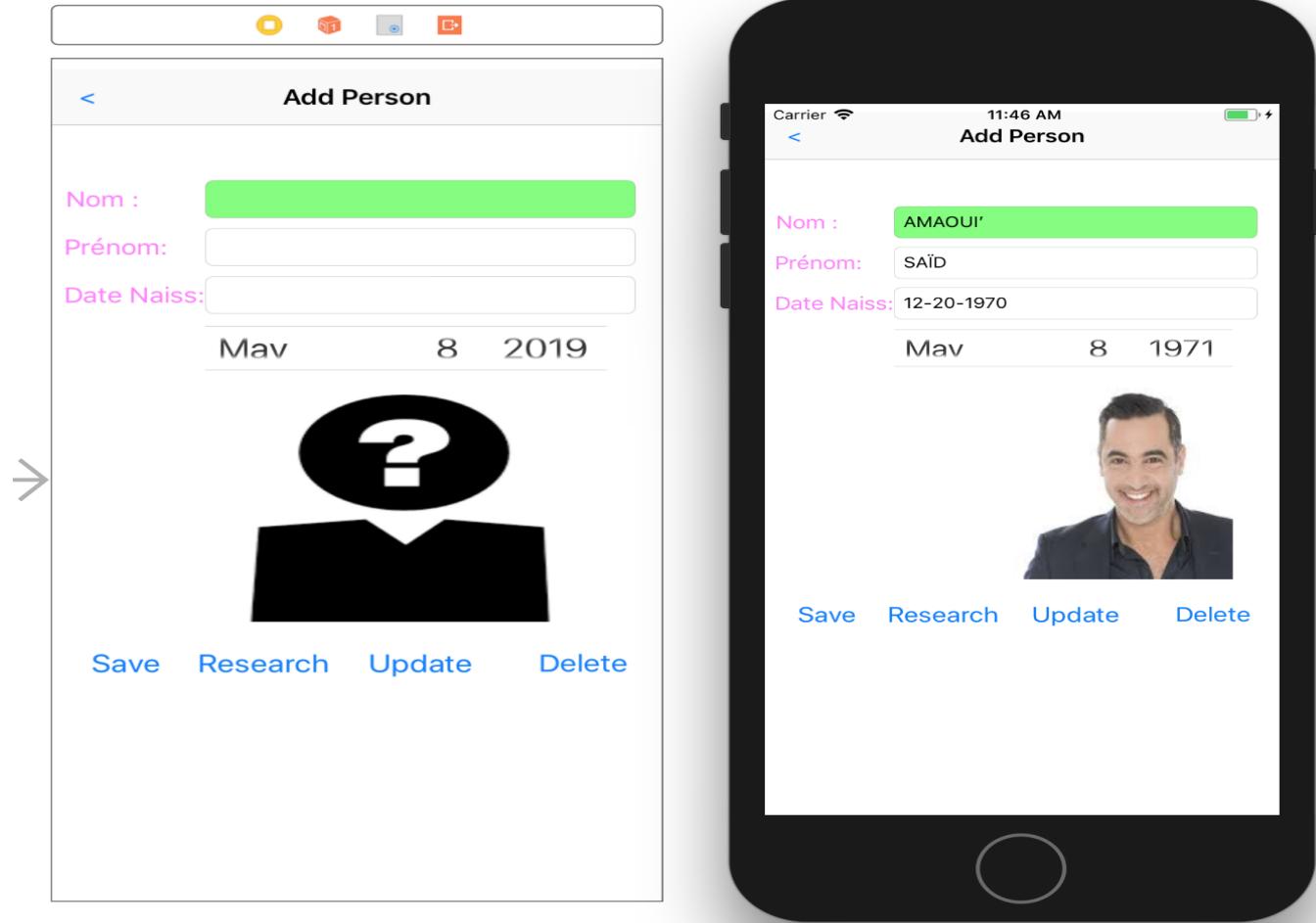


# **NSPersistentContainer**

Permet de réduire l'installation de **Core Data** à seulement trois étapes :

1. Créer un fichier **DataModel.xcdatamodeld**.
2. Instancier **NSPersistentContainer** en lui passant : le nom du fichier "DataModel".
3. Récupérer un contexte à partir de l'instance de **NSPERSISTANTContainer**

# App : qui gère l'ensemble des étudiants d'une classe donnée



# Récupération du contexte des objets de gestion



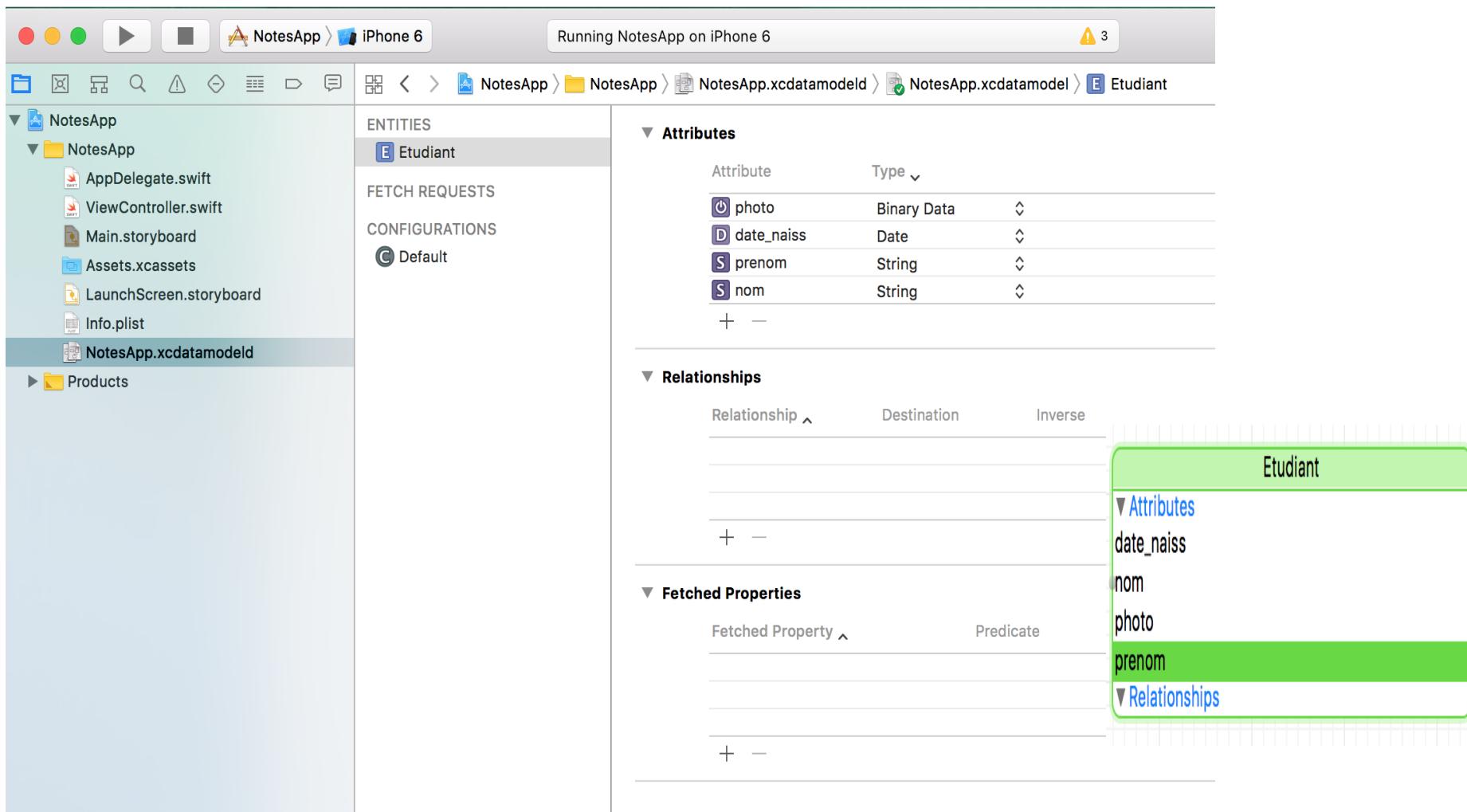
The screenshot shows the Xcode interface with the following details:

- Toolbar:** Standard Xcode icons for file operations like Open, Save, Find, and Build.
- Project Navigator:** Shows the project structure:
  - A folder named "niveau3".
  - A folder named "niveau3" containing a "model" folder.
  - A file named "AppDelegate.swift" which is selected and highlighted in green.
  - Other files listed include "ViewController.swift", "DetailViewController.swift", "TableViewCell.swift", "Main.storyboard", and "Assets.xcassets".
- Search Bar:** Displays the path: "niveau3 > niveau3 > AppDelegate.swift" and the method name "application(\_:didFinishLaunchingWithOptions:)".
- Editor:** Displays the Swift code for the AppDelegate class:

```
1
2
3 import UIKit
4 import CoreData
5
6 let ad : AppDelegate = UIApplication.shared.delegate as! AppDelegate
7 let context = ad.persistentContainer.viewContext
8
```

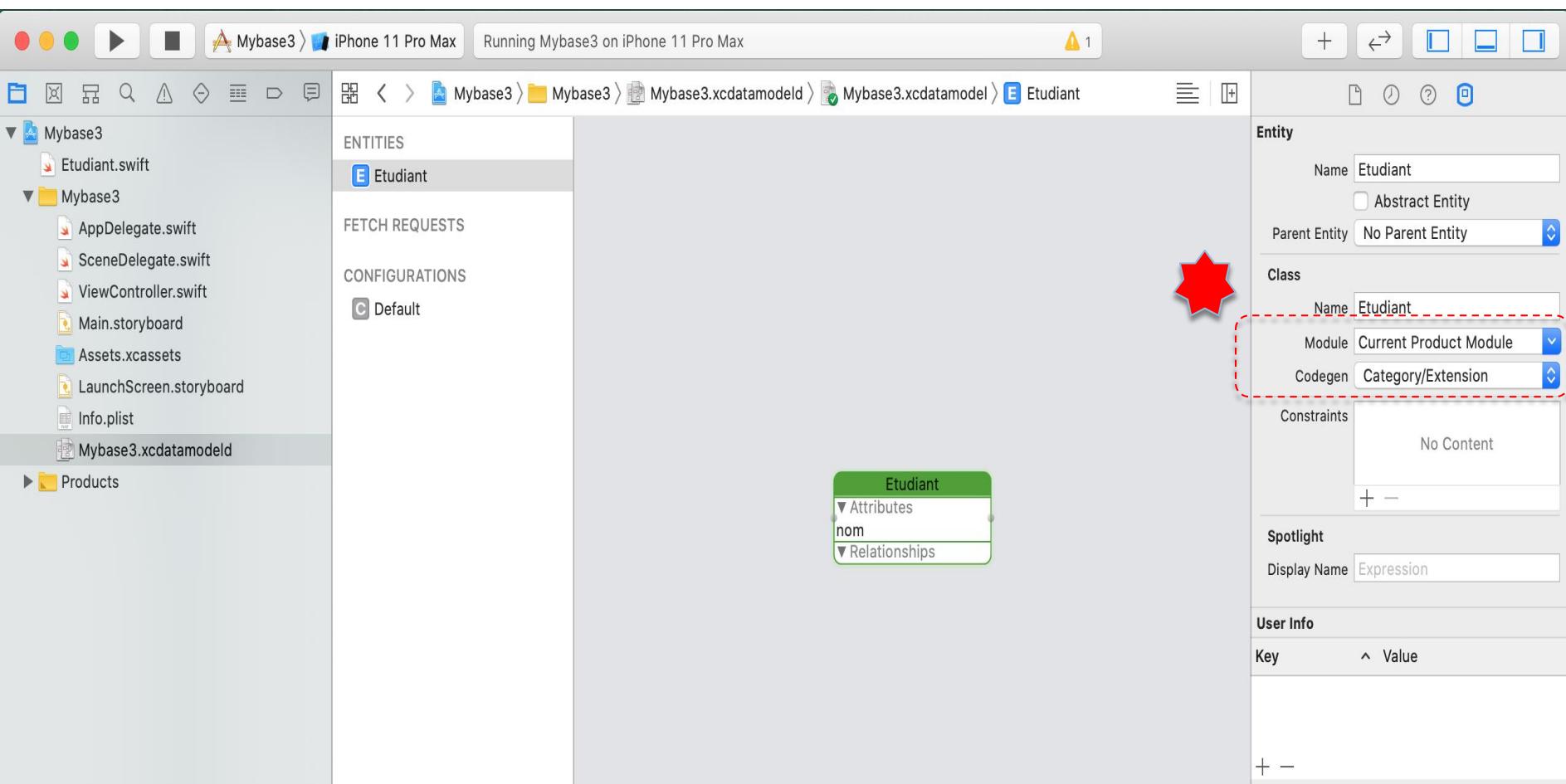
# Stockage persistants : CoreData

- 1 : Créer un projet avec l'option CoreData
- 2 : importer le framework CoreData dans le code
- 3 : Créer une entité Etudiant avec les champs : Nom ,Age et photo



# Stockage persistants : CoreData

- 1 : Créer un projet avec l'option CoreData
- 2 : importer le framework CoreData dans le code
- 3 : Créer une entité Etudiant avec les champs : Nom ,Age et photo



# Stockage persistants : Sauvegarde

## □ Récupérer le contexte de l' application par :

- let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
- let context : NSManagedObjectContext = appDel.persistentContainer.viewContext

## □ Récupérer la description de l'entité à traiter :

let newEtudiant = **Etudiant(context)**

## □ Inserer un Etudiant :

- let v = UIImagePNGRepresentation(UIImage(named: "cloud\_small")!)
- newEtudiant.nom = "Farah"
- newEtudiant.date\_naiss = formater.date(from: "12-01-1978")
- newEtudiant.photo = V

## □ Sauvegarder le contexte avec traitement des exceptions:

```
do {  
    try context.save()  
}
```

# Stockage persistants : Sauvegarde

## □ Récupérer le contexte de l' application par :

- let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
- let context : NSManagedObjectContext = appDel.persistentContainer.viewContext

## □ Récupérer la description de l'entité à traiter :

```
let newEtudiant = NSEntityDescription.insertNewObject(forEntityName: "Etudiant", into: context)
```

## □ Insérer un Etudiant :

- let v = UIImagePNGRepresentation(UIImage(named: "cloud\_small")!)
- newEtudiant.setValue("farah", forKey: "nom")
- newEtudiant.setValue(293, forKey: "age")
- newEtudiant.setValue(v, forKey: "photo")



Deuxième  
façon pour  
insérer un  
étudiant

## □ Sauvegarder le contexte avec traitement des exceptions:

```
do {  
    try context.save()  
}
```

# Fonction : Sauvegarde

```
import CoreData
import UIKit
class ViewController:
    UIViewController, UINavigationControllerDelegate, UIImagePickerControllerDelegate {
    @IBOutlet weak var nom_txt: UITextField!
    @IBOutlet weak var prenom_txt: UITextField!
    @IBOutlet weak var photo: UIImageView!
    @IBOutlet weak var date_naiss_text: UITextField!
    let formater = DateFormatter()
    let pkr_image = UIImagePickerController()

    @IBAction func save(_ sender: UIButton) {
        let newEtudiant = Etudiant(context: context)
        newEtudiant.nom = nom_txt.text
        newEtudiant.prenom = prenom_txt.text
        formater.dateFormat = "MM-dd-YYYY"
        newEtudiant.date_naiss = formater.date(from: date_naiss_text.text!)

        newEtudiant.photo = UIImagePNGRepresentation(photo.image!)
        do {
            ad.saveContext()
            print("saved")
        }
    }
}
```

# Stockage persistants : Affichage

## Récupérer le contexte de l' application par :

- let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
- let context : NSManagedObjectContext = appDel.persistentContainer.viewContext

## □ Chercher l'étudiants "balouki" par la requête:

```
let request = NSFetchedRequest<Etudiant>(entityName: "Etudiant")  
request.predicate = NSPredicate(format: "nom = %@", nom_text....,
```

Nom :

balouki

## □ Afficher l'étudiants "balouki" :

```
let resultat = try context.fetch(request)  
if !resultat.isEmpty {  
    for itm in resultat {  
        nom_txt.text = itm.nom!  
        prenom_txt.text = itm.prenom  
        formater.dateFormat = "MM-dd-YYYY"  
        date_naiss_text.text = formater.string(from: itm.date_naiss!)  
        photo.image = UIImage(data : itm.photo as! Data)  
    }  
}
```

# Fonction : Afficher

```
@IBAction func lister(_ sender: UIButton) {
    let request = NSFetchedResultsController<Etudiant>(entityName :"Etudiant")
    request.predicate = NSPredicate(format: "nom = %@", nom_txt.text!)
    do {
        let rsl = try context.fetch(request)
        if !rsl.isEmpty {
            for itm in rsl {
                nom_txt.text = itm.nom!
                prenom_txt.text = itm.prenom
                formater.dateFormat = "MM-dd-YYYY"
                date_naiss_text.text = formater.string(from: itm.date_naiss!)
                photo.image = UIImage(data : itm.photo as! Data) ⚠️ Forced cast from
            }
        }
        else{
            vider_champs()
        }
    } catch {
        print("Erreur d'affichage")
    }
}
```

# Stockage persistants : Modifier

## □ Récupérer le contexte de l' application par :

- let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
- let context : NSManagedObjectContext = appDel.persistentContainer.viewContext

## □ Chercher l'étudiants “balouki” par la requête:

```
let request = NSFetchedRequest<Etudiant>(entityName: "Etudiant")
request.predicate = NSPredicate(format: "nom = %@", »nom_text.text")
```

## □ Modifier l'étudiants “balouki” :

```
let rsl = try context.fetch(request)
for itm in rsl {
    itm.setValue(nom_txt.text, forKey: "nom")
    itm.setValue(prenom_txt.text, forKey: "prenom")
    formater.dateFormat = "MM-dd-YYYY"
    itm.setValue(formater.date(from: date_naiss_text.text!), forKey: "date_naiss")
    itm.setValue(UIImagePNGRepresentation(photo.image! ), forKey: "photo")
```

# Fonction : Modification

```
@IBAction func update(_ sender: UIButton) {

    let request = NSFetchedResultsController<Etudiant>(entityName :"Etudiant")
    request.predicate = NSPredicate(format: "nom = %@", nom_txt.text!)
    do {
        let rsl = try context.fetch(request)

        for itm in rsl {
            print("-----start update-----")
            itm.setValue(nom_txt.text, forKey: "nom")
            itm.setValue(prenom_txt.text, forKey: "prenom")
            formater.dateFormat = "MM-dd-YYYY"
            itm.setValue(formater.date(from: date_naiss_text.text!), forKey: "date_naiss")
            itm.setValue(UIImagePNGRepresentation(photo.image!) ), forKey: "photo")
            do {
                try context.save()
            }
            print("-----end update-----")
        }
    } catch {
        print("Erreur ")
    }
}
```

# Stockage persistants : Supprission

- Récupérer le contexte de l' application par :
- let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
- let context : NSManagedObjectContext = appDel.persistentContainer.viewContext
- Récupérer l'ensemble des étudiants par la requete:

```
let request = NSFetchedRequest<Etudiant>(entityName: "Etudiant")
```

- Suprimer l'étudiant portant le nom balouki:
- context.delete(rst as NSManagedObject)
- Sauvgarder le contexte avec traitement des exceptions:

```
do {  
    try context.save()  
}
```

# Fonction : Supprimer

```
@IBAction func supprimer(_ sender: UIButton) {
    let request = NSFetchedResultsController<Etudiant>(entityName :"Etudiant")
    // fonction qui cherche l'étudiant dont le nom est saisi dans textField : nom_txt
    //puis supprime cet étudiant de la base de donnée
    request.predicate = NSPredicate(format: "nom = %@", nom_txt.text!)
    do {
        let rsl = try context.fetch(request)
        for itm in rsl {
            print("-----start delete-----")
            context.delete(itm)
            do {
                try context.save()
            }
            print("-----end delete-----")
        }
        vider_champs()
    } catch {
        print("Erreur ")
    }
}
```

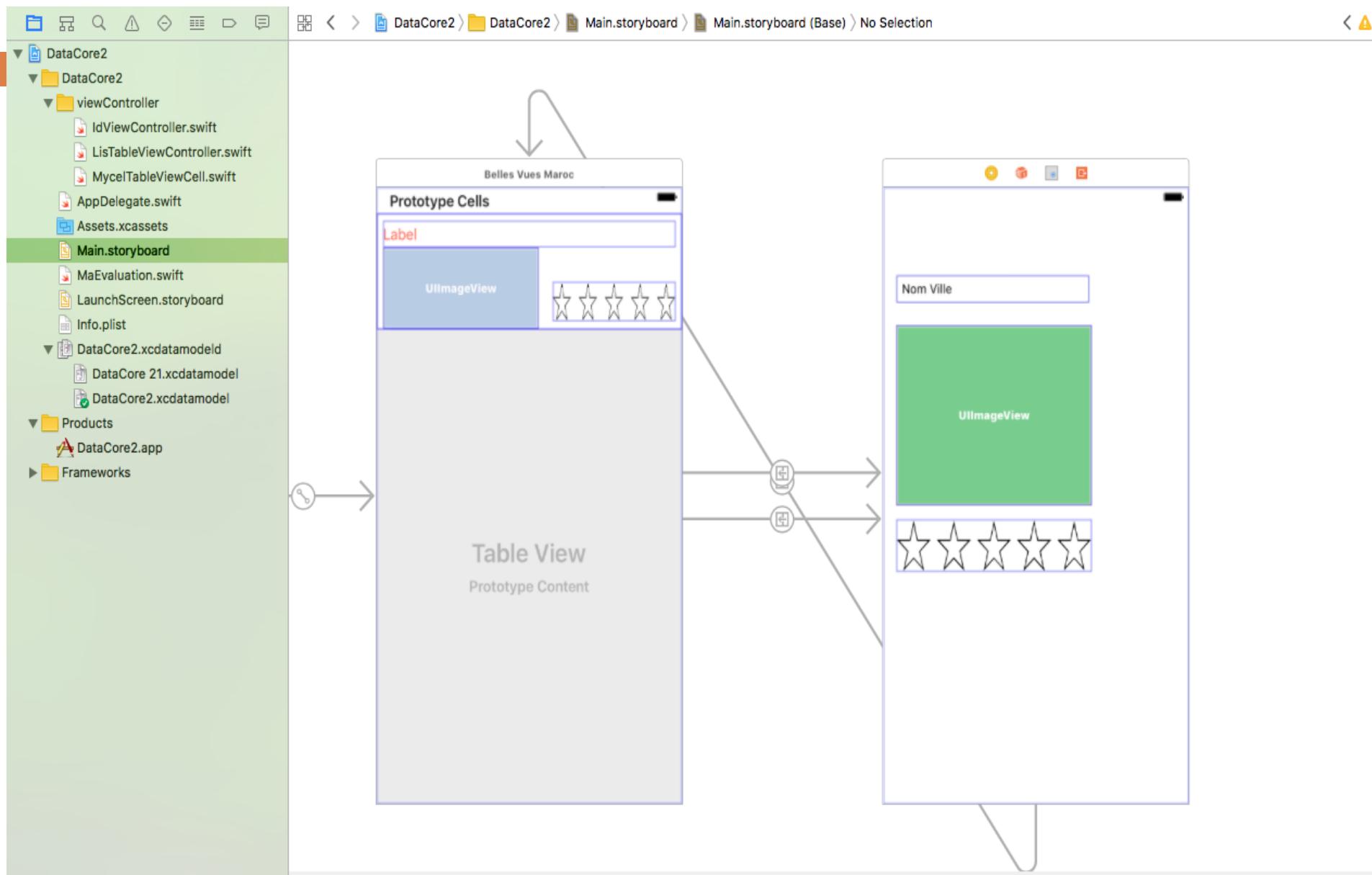
# Stockage persistants : CoreData

## TP :

Une application qui permet de défiller les belles vues de chaque ville avec Contrôle de vote.

1. Création d'un nouveau projet de type CoreData
2. Ajouter ds la vue principale une ListView
3. Création d'une classe de type :UITableViewCell
4. Etablir le lien entre la nouvelle classe et la cellule de ListView
5. Ajouter une vue Secondaire qui contient :  
Label : Nom ville,  
Image : belle Vue (PickerImage + reconnaissance de geste) **réalisé précédemment**  
HorizontalStackView : Zone d'évaluation **réalisé précédemment**
6. Etablir des segues entre la vue principale et secondaire
  - Segue pour ajouter une nouveau enregistrement
  - Segue pour modifier un enregistrement
  - segue pour terminer
- Gérer les M-à-J des données d'une manière permanente : CoreData

# TP : Structure de projet



# TP : Création de l'Entité Nature

The screenshot shows the Xcode interface with the DataCore2 project open. On the left, the Project Navigator displays the following structure:

- DataCore2
- ↳ DataCore2
- ↳ viewController
  - ↳ IdViewController.swift
  - ↳ ListTableVi...roller.swift
  - ↳ MycelTabl...wCell.swift
- ↳ AppDelegate.swift
- ↳ Assets.xcassets
- ↳ Main.storyboard
- ↳ MaEvaluation.swift
- ↳ LaunchScreen.storyboard
- ↳ Info.plist
- ↳ DataCore2.xcdatamodeld
  - ↳ DataCore...cdatamodel
  - ↳ DataCore2...datamodel
- ↳ Products
  - ↳ DataCore2.app
- ↳ Frameworks

In the center, the Data Model Inspector shows the "ENTITIES" section with "Nature" selected. Below it are sections for "FETCH REQUESTS" and "CONFIGURATIONS".

On the right, the main workspace displays the "Nature" entity in the Core Data editor. The entity has the following attributes:

- evaluation
- nom
- photo

A "Relationships" section is also present.

# TP : Definition de la classe : MycelTableViewCell

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with the following files:
  - DataCore2
  - DataCore2
  - viewController
    - IdViewController.swift
    - LisTableVi...troller.swift
    - MycelTabl...wCell.swift
  - AppDelegate.swift
  - Assets.xcassets
  - Main.storyboard
  - MaEvaluation.swift
  - LaunchScreen.storyboard
  - Info.plist
- File Navigator:** Shows the file path: DataCore2 > DataCore2 > viewController > MycelTableViewCell.swift
- Editor:** Displays the code for `MycelTableViewCell`:

```
import UIKit

class MycelTableViewCell: UITableViewCell {

    @IBOutlet weak var nom: UILabel!

    @IBOutlet weak var evalution: MaEvaluation!
    @IBOutlet weak var photo: UIImageView!
}
```

# TP : Structure de la Vue Principale

The screenshot shows the Xcode interface with the project 'DataCore2' open. The left sidebar displays the file structure, including 'viewController' and 'ListTableVi...troller.swift'. The main editor window shows the code for 'ListTableViewController.swift'.

```
import UIKit
import CoreData
class ListTableViewController: UITableViewController {
    var list = [Nature]()
    override func viewDidLoad(_ animated: Bool) {
        let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
        let context : NSManagedObjectContext = appDel.persistentContainer.viewContext
        let req = NSFetchedResultsController<Nature>(entityName: "Nature", fetchRequest: nil, managedObjectContext: context, sectionNameKeyPath: nil)
        do { list = try context.fetch(req) }
        catch { fatalError( error as! String) }
        tableView.reloadData()
    }
    override func numberOfSections(in tableView: UITableView) -> Int {
        return list.count
    }
    override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) as! MyTableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "MyCell", for: indexPath) as! MyTableViewCell
        cell.nameLabel.text = list[indexPath.row].name
        cell.imageView.image = UIImage(named: list[indexPath.row].image)
        return cell
    }
}
```

The right side of the screen shows the running iOS application on an iPhone 6 simulator. The title bar reads 'Belles Vues Maroc'. The application displays a list of travel destinations with images and star ratings:

- rabat**: Image of a tree, 1 star
- Casablanca star**: Image of a waterfall, 1 star
- cd**: Image of a waterfall, 3 stars
- Agadir**: Image of a waterfall, 4 stars
- béni l'Elmaleh**: Image of a waterfall, 5 stars

# TP : Vue Principale

The screenshot shows the Xcode interface with the project navigation bar at the top. The left sidebar displays the project structure for 'DataCore2'. The main editor area shows the Swift code for 'LisTableViewController.swift'.

```
import CoreData
class LisTableViewController: UITableViewController {
    var list = [Nature]()
    override func viewDidAppear(_ animated: Bool) {
        let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
        let context : NSManagedObjectContext = appDel.persistentContainer.viewContext
        let req = NSFetchedResultsController<Nature>(entityName: "Nature")
        do { list = try context.fetch(req) }
        catch { fatalError( error as! String) }
        tableView.reloadData()
    }
    override func numberOfSections(in tableView: UITableView) -> Int { return 1 }
    override func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return list.count
    }
    override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = tableView.dequeueReusableCell(withIdentifier: "reuseIdentifier", for: indexPath) as! MycelTableViewCell
        cell.nom.text = list[indexPath.row].nom!
        cell.evaluation.evaluation = Int(list[indexPath.row].evaluation)
        if list[indexPath.row].photo != nil {
            cell.photo.image = UIImage(data: list[indexPath.row].photo as! Data)
        }
        return cell
    }
}
```

# TP : Vue Principale

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure under "DataCore2". The file "LisTableViewController.swift" is selected.
- Editor:** Displays the Swift code for "LisTableViewController.swift".
- Callouts:** A callout box highlights the word "override" in the first method definition, with options: "Step into", "Step into instruction (hold Control)", and "Step into thread (hold Control-Shift)".
- Bottom Bar:** Includes standard Xcode navigation icons like back, forward, and search.

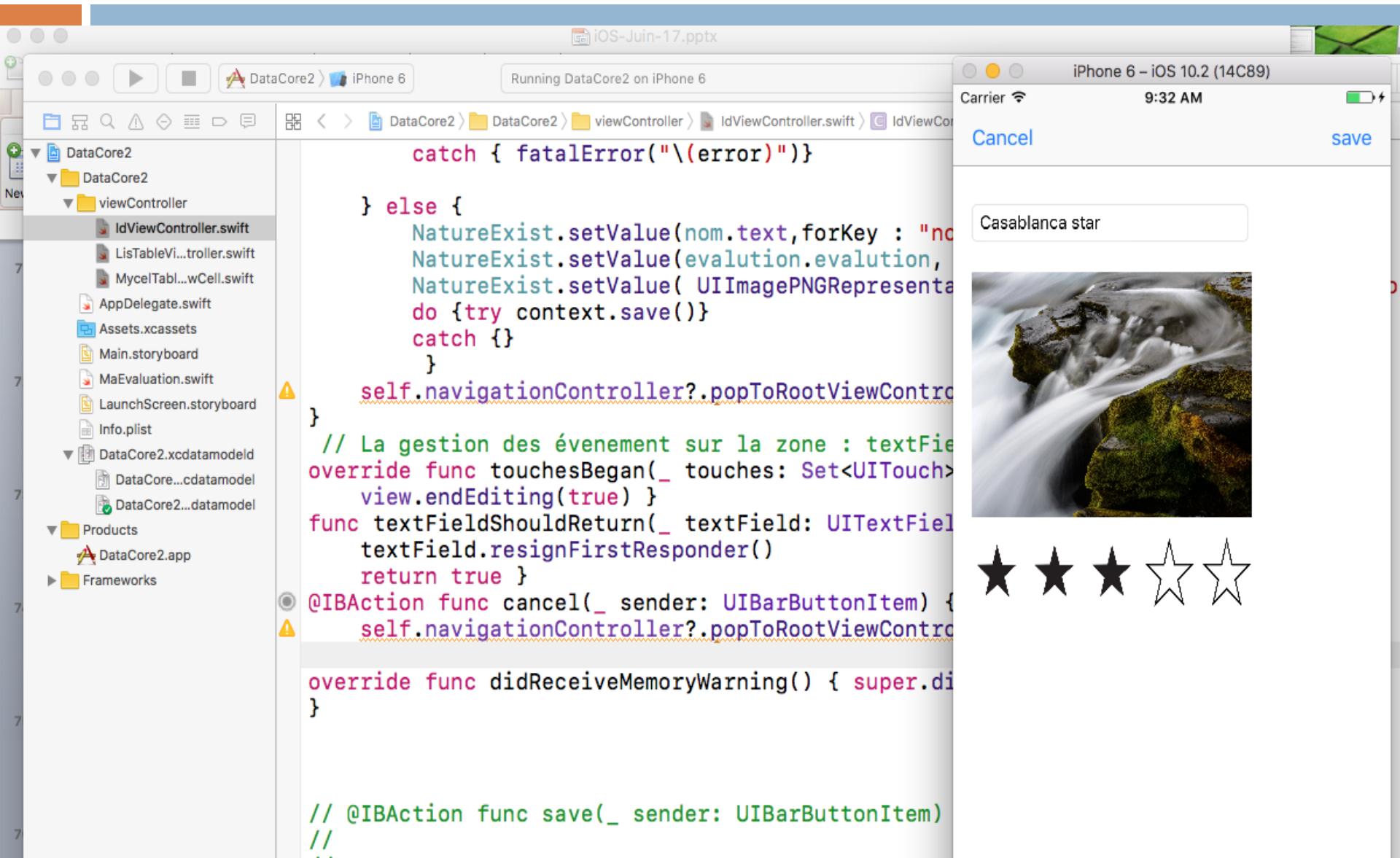
```
override func tableView(_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {
    if editingStyle == .delete {
        let appDel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
        let context : NSManagedObjectContext = appDel.persistentContainer.viewContext
        context.delete(list[indexPath.row] as NSManagedObject)
        do {try context.save()}
        catch {}

        list.remove(at: indexPath.row)
        tableView.deleteRows(at: [indexPath], with: .fade)
    }
}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "byby" {
        let ss = segue.destination as! IdViewController
        ss.NatureExist = list[(tableView.indexPathForSelectedRow?.row)!]
    }
}

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

# TP : Structure de la Vue Secondaire



```
    catch { fatalError("\(error)")}

} else {
    NatureExist.setValue(nom.text(forKey: "name"), forKey: "name")
    NatureExist.setValue(evolution.evalution, forKey: "evalution")
    NatureExist.setValue(UIImagePNGRepresentation(image.image?.cgImage), forKey: "image")
    do {try context.save()}
    catch {}
}
self.navigationController?.popToRootViewController(animated: true)

// La gestion des événements sur la zone : textFieldShouldReturn
override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
    view.endEditing(true)
}
func textFieldShouldReturn(_ textField: UITextField) -> Bool {
    textField.resignFirstResponder()
    return true
}
@IBAction func cancel(_ sender: UIBarButtonItem) {
    self.navigationController?.popToRootViewController(animated: true)
}

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

// @IBAction func save(_ sender: UIBarButtonItem) {
// }
```

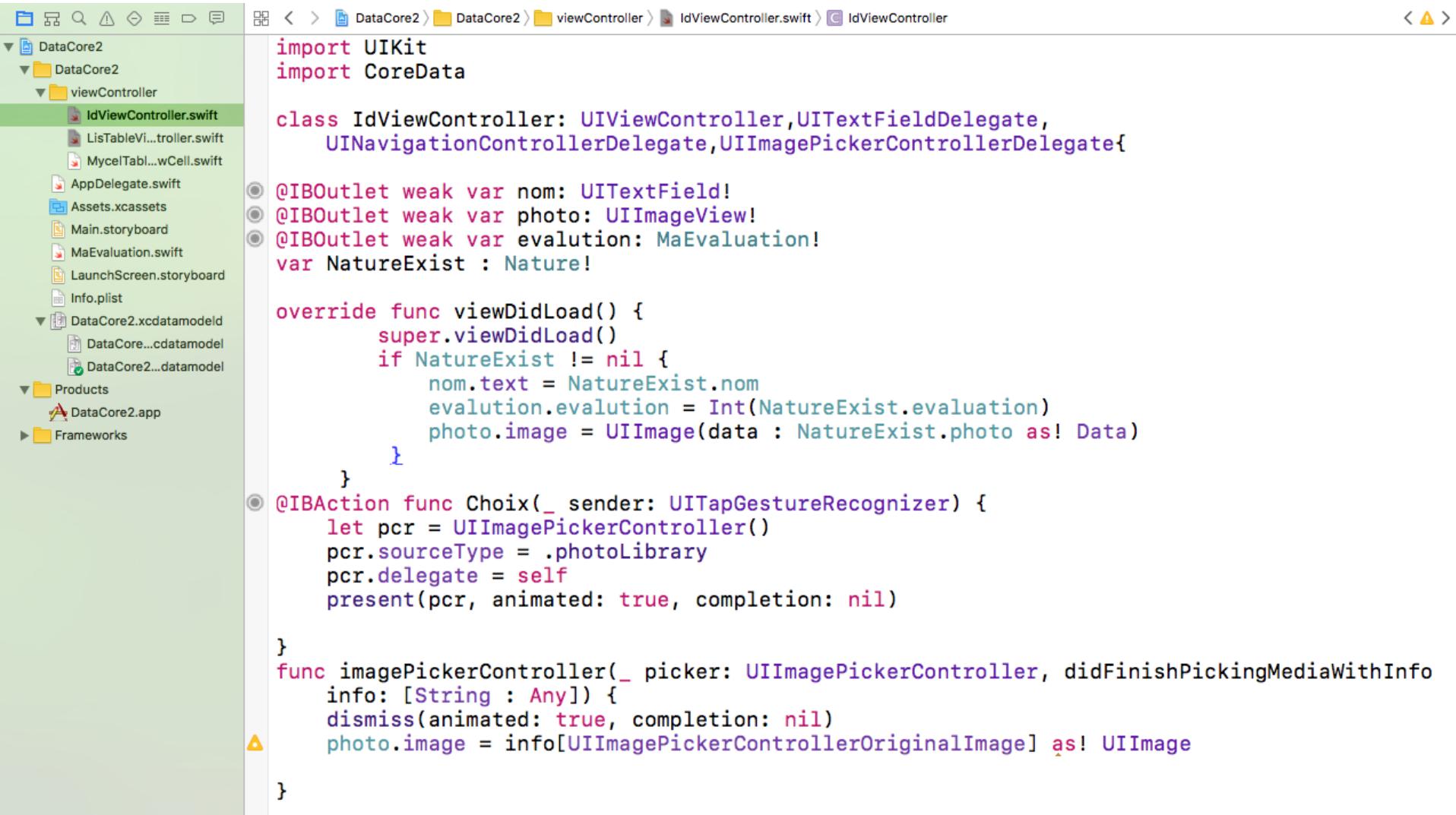
Cancel

Casablanca star



★ ★ ★ ★ ★

# TP : Vue Secondaire



The screenshot shows the Xcode interface with the project navigation bar at the top. Below it is the file browser on the left, showing the project structure:

- DataCore2
- DataCore2
- viewController
- IdViewController.swift
- ListTableVi...roller.swift
- MycelTabl...wCell.swift
- AppDelegate.swift
- Assets.xcassets
- Main.storyboard
- MaEvaluation.swift
- LaunchScreen.storyboard
- Info.plist
- DataCore2.xcdatamodeld
- DataCore...cdatamodel
- DataCore2...datamodel
- Products
- DataCore2.app
- Frameworks

The IdViewController.swift file is selected in the file browser. The main editor area contains the following Swift code:

```
import UIKit
import CoreData

class IdViewController: UIViewController, UITextFieldDelegate,
    UINavigationControllerDelegate, UIImagePickerControllerDelegate{

    @IBOutlet weak var nom: UITextField!
    @IBOutlet weak var photo: UIImageView!
    @IBOutlet weak var evalution: MaEvaluation!
    var NatureExist : Nature!

    override func viewDidLoad() {
        super.viewDidLoad()
        if NatureExist != nil {
            nom.text = NatureExist.nom
            evalution.evaluation = Int(NatureExist.evaluation)
            photo.image = UIImage(data : NatureExist.photo as! Data)
        }
    }

    @IBAction func Choix(_ sender: UITapGestureRecognizer) {
        let pcr = UIImagePickerController()
        pcr.sourceType = .photoLibrary
        pcr.delegate = self
        present(pcr, animated: true, completion: nil)
    }

    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
        dismiss(animated: true, completion: nil)
        photo.image = info[UIImagePickerControllerOriginalImage] as! UIImage
    }
}
```

# TP : Vue Secondaire



The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with files like DataCore2.swift, ViewController.swift, AppDelegate.swift, Assets.xcassets, Main.storyboard, Info.plist, DataCore2.xcdatamodeld, Products, and DataCore2.app.
- Editor:** Displays Swift code for a secondary view controller. The code handles saving data to Core Data and navigating back to the root view controller.
- Issues:** A warning icon is present near the self.navigationController?.popToRootViewController(animated: true) line.

```
    @IBOutlet weak var save: UIBarButtonItem!
    @IBAction func save(_ sender: UIBarButtonItem) {
        let appldel : AppDelegate = UIApplication.shared.delegate as! AppDelegate
        let context : NSManagedObjectContext = appldel.persistentContainer.viewContext
        if NatureExist == nil {
            let newNature = NSEntityDescription.insertNewObject(forEntityName: "Nature", into: context)

            newNature.setValue(nom.text, forKey: "nom")
            newNature.setValue(evolution.evalution, forKey: "evaluation")
            if photo.image != nil {
                newNature.setValue( UIImagePNGRepresentation(photo.image!), forKey: "photo")
            }
            do { try context.save()
            } catch { fatalError("\(error)")}

        } else {
            NatureExist.setValue(nom.text, forKey: "nom")
            NatureExist.setValue(evolution.evalution, forKey: "evaluation")
            NatureExist.setValue( UIImagePNGRepresentation(photo.image!), forKey: "photo")
            do {try context.save()
            } catch {}
        }
        self.navigationController?.popToRootViewController(animated: true)
    }
    // La gestion des événement sur la zone : textField
    override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
        view.endEditing(true)
    }
    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder()
        return true
    }
    @IBAction func cancel(_ sender: UIBarButtonItem) {
        self.navigationController?.popToRootViewController(animated: true)
    }
}
```

# Géolocalisation : CoreLocation

1: option “Build Phases” permet de loader le framework : CoreLocation

The screenshot shows the Xcode interface with the project 'ouSuisJe' selected. The left sidebar lists files like AppDelegate.swift, ViewController.swift, and Main.storyboard. The main area is the 'Build Phases' tab, which is currently active. It shows the following sections:

- Target Dependencies (0 items)**
- Compile Sources (3 items)**
- Link Binary With Libraries (2 items)**

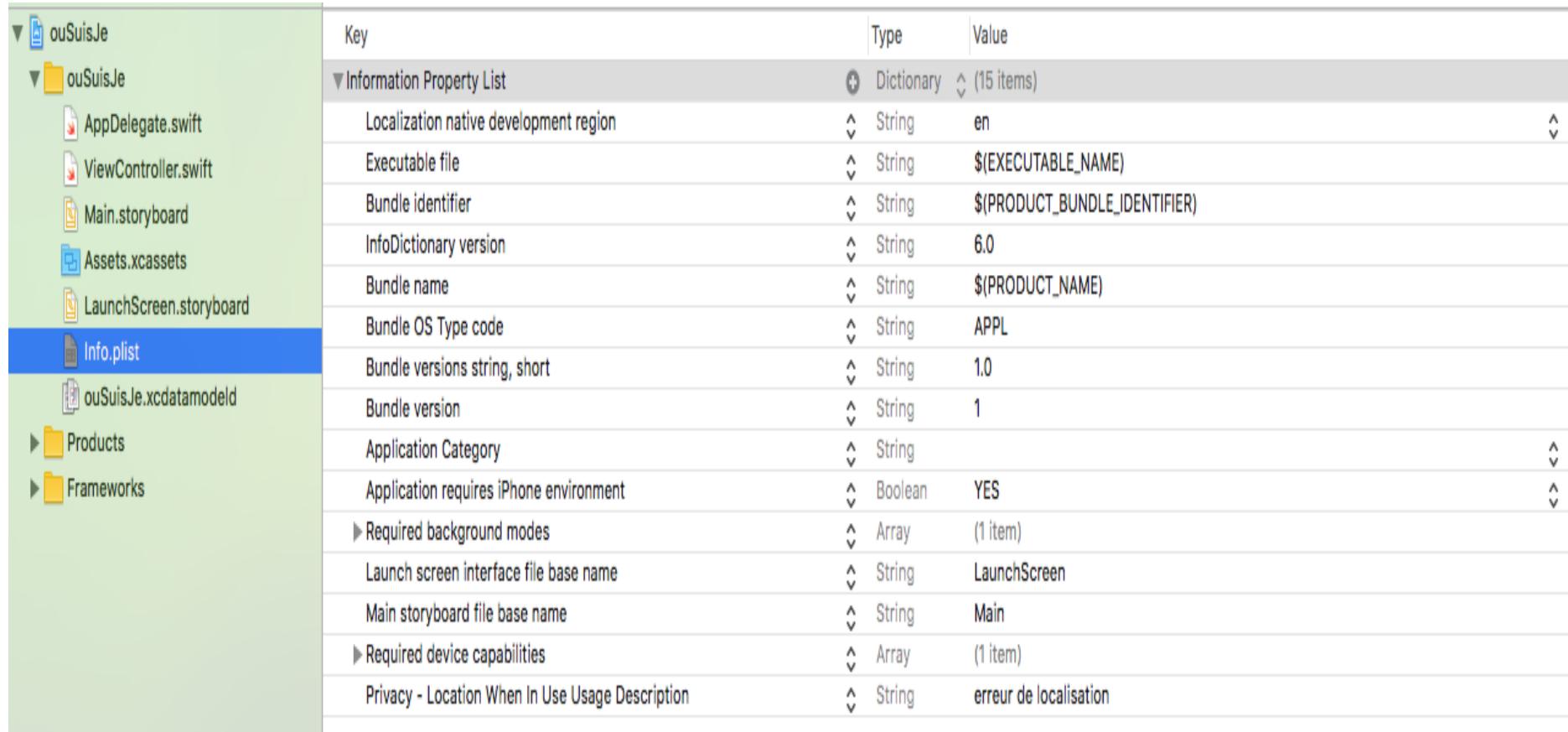
Name	Status
MapKit.framework	Required ▲
CoreLocation.framework	Required ▼

Below this section are buttons for adding (+) or removing (-) frameworks, and a note: "Drag to reorder frameworks".

- Copy Bundle Resources (3 items)**

# Géolocalisation : CoreLocation

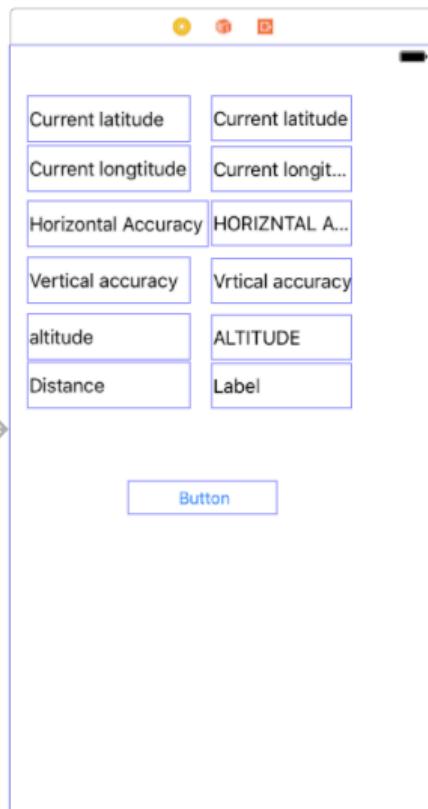
## 2 : définir un message d'erreur dans la propriété : Location when use usage description



The screenshot shows the Xcode interface with a project named "ouSuisJe". On the left, the file structure is visible, including "Info.plist" which is currently selected and highlighted in blue. The main area displays the contents of "Info.plist" as a table.

Key	Type	Value
▼ Information Property List	Dictionary	(15 items)
Localization native development region	String	en
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application Category	String	
Application requires iPhone environment	Boolean	YES
► Required background modes	Array	(1 item)
Launch screen interface file base name	String	LaunchScreen
Main storyboard file base name	String	Main
► Required device capabilities	Array	(1 item)
Privacy - Location When In Use Usage Description	String	erreur de localisation

# Géolocalisation : CoreLocation



```
import UIKit
import CoreLocation
import MapKit

class ViewController: UIViewController, CLLocationManagerDelegate {
    @IBOutlet weak var map: MKMapView!

    @IBOutlet weak var latitude: UILabel!
    @IBOutlet var label: UILabel!
    @IBOutlet weak var longitude: UILabel!
    @IBOutlet weak var vaccuracy: UILabel!

    @IBOutlet weak var altitude: UILabel!
    @IBOutlet weak var HAccuracy: UILabel!
    var locationManager: CLLocationManager = CLLocationManager()
    var startLocation: CLLocation!

    @IBAction func reset(_ sender: Any) {
        startLocation = nil // initialise le point de départ à nil
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        // permet de choisir la précision
        locationManager.desiredAccuracy = kCLLocationAccuracyBest
        locationManager.delegate = self

        // demande l'autorisation d'utiliser les ressources
        locationManager.requestWhenInUseAuthorization()
        locationManager.startUpdatingLocation()
        // le point de départ a la valeur nil au moment de chargemnt
        startLocation = nil
    }
}
```

# Géolocalisation : CoreLocation



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

func locationManager(_ manager: CLLocationManager, didUpdateLocations
locations: [CLLocation]) {

    // lorsque on change de position, on peut récupérer latitude, longitude....
    let latestLocation: CLLocation = locations[locations.count - 1]
    latitude.text = String(format: "%.4f", latestLocation.coordinate.latitude)
    longitude.text = String(format: "%.4f", latestLocation.coordinate.longitude)
    HAccuracy.text = String(format: "%.4f", latestLocation.horizontalAccuracy)
    altitude.text = String(format: "%.4f", latestLocation.altitude)
    vaccuracy.text = String(format: "%.4f", latestLocation.verticalAccuracy)

    if startLocation == nil {
        startLocation = locations[locations.count - 1]
    }
    let distanceBetween: CLLocationDistance = latestLocation.distance(from:
        startLocation)
    label.text = String(format: "%.2f", distanceBetween)
}

func locationManager(_ manager: CLLocationManager, didFailWithError error:
    Error) {
    // pour traiter les erreurs de localisations
}

}
```

# Géolocalisation : CoreLocation

The screenshot shows the Xcode interface with a storyboard on the left and a Swift code editor on the right.

**Storyboard:** Displays a view controller with several outlets: Current latitude, Current longitude, Horizontal Accuracy, Vertical accuracy, altitude, Distance, and a Button.

**Code Editor (ViewController.swift):**

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

func locationManager(_ manager: CLLocationManager,
didUpdateLocations locations: [CLLocation]) {
    // lorsque on change de position, on peut récupérer
    // latitude, longitude....
    let latestLocation: CLLocation = locations[locations.
        count - 1]
    latitude.text = String(format: "%.4f", latestLocation.
        coordinate.latitude)
    longitude.text = String(format: "%.4f", latestLocation.
        coordinate.longitude)
    HAccuracy.text = String(format: "%.4f", latestLocation.
        horizontalAccuracy)
    altitude.text = String(format: "%.4f", latestLocation.
        altitude)
    vaccuracy.text = String(format: "%.4f", latestLocation.
        verticalAccuracy)

    if startLocation == nil {
        startLocation = locations[locations.count - 1]
    }
    let distanceBetween: CLLocationDistance =
        latestLocation.distance(from: startLocation)
    label.text = String(format: "%.2f", distanceBetween)
}
```

**Simulator Output (iPhone 6 - iOS 10.2):**

	Value
Current latitude	37.3377
Current longitude	-122.0300
Horizontal Accuracy	10.0000
Vertical accuracy	-1.0000
altitude	0.0000
Distance	601.99

# Maps : Utilisation de MapKit

The screenshot shows the Xcode interface with a project named "map\_test1". A modal dialog titled "Choose frameworks and libraries to add:" is open, displaying a search bar and a list of available frameworks under "iOS 8.3". The "MapKit.framework" is selected and highlighted with a blue background. At the bottom of the dialog are three buttons: "Add Other...", "Cancel", and "Add". The "Add" button is highlighted in blue.

map\_test1.xcodeproj

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help

map\_test1: Ready | Today at 23:16

Build Rules

Identity and Type

Name map\_test1

Location Absolute

Full Path /Users/ybalouki/Desktop/gg/map\_test1/map\_test1.xcodeproj

Project Document

Project Format Xcode 3.2-compatible

Organization youssef balouki

Class Prefix

Text Settings

Indent Using Spaces

Widths 4 Tab Indent

Wrap lines

Source Control

Repository gg

Type Git

Current Branch Branches Unavailable

Add Other... Cancel Add

Build phases/Link Binary With Libraries/+/cherche map

Youssef BALOUKI

# Maps : Utilisation de MapKit

The screenshot shows the Xcode interface with the storyboard and code editor open.

**Main.storyboard:** Displays a single **Map View** component. A control center panel is visible on the left, showing the **Map View** selected. The storyboard navigation bar indicates "map\_test1 > iPhone 6".

**ViewController.swift:** The code defines a **ViewController** class conforming to **UIViewController** and **MKMapViewDelegate**. It overrides **viewDidLoad()** and **didReceiveMemoryWarning()**.

```
// ViewController.swift
// map_test1
//
// Created by youssef balouki on 30/03/16.
// Copyright (c) 2016 youssef balouki. All rights reserved.

import UIKit
import MapKit

class ViewController: UIViewController, MKMapViewDelegate {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

Running map\_test1 on iPhone 5

```
// ViewController.swift
// map_test1
//
// Created by youssef balouki on 30/03/16.
// Copyright (c) 2016 youssef balouki. All rights reserved.

import UIKit
import MapKit

class ViewController: UIViewController, MKMapViewDelegate {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



google.fr

Connexion

fstsett

Restaurant universitaire

Ecole Nationale de Commerce et de Gestion

Faculté des Sciences et Techniques

4,2 ★★★★☆ 20 avis

Université

ENREGISTRER RECHERCHER À PROXIMITÉ ENVOYER VERS VOTRE TÉLÉPHONE PARTAGER

BP : 577, Route de Casa -, Settat, Maroc  
+212 671-332853  
Revendiquer cet établissement  
Suggérer une modification

Ajouter des informations manquantes

Ajouter un site Web

5 photos

Satellite

Settat Province Maroc 33.031024, -7.616685

Transports en commun Cliquez sur les arrêts pour en savoir plus.

Données cartographiques ©2016 Google Conditions Confidentialité Envoyer des commentaires 100 m

The screenshot shows a Google Maps interface. On the left, there's a sidebar for 'fstsett' with a photo of a university building, a rating of 4.2 from 20 reviews, and links to register, search nearby, share, and add a website. Below that are contact details: address (BP 577, Route de Casa, Settat, Maroc), phone number (+212 671-332853), claim the business, and suggest edits. There are also buttons to add missing information and add a website. At the bottom left, there are five thumbnail images of the building. The main area is a map of the university campus. It features several buildings: 'Ecole Nationale de Commerce et de Gestion' (highlighted in yellow), 'Faculté des Sciences et Techniques' (highlighted in red), and 'Restaurant universitaire' (highlighted in orange). Other labeled areas include 'cafe' and 'Crepuscule Club'. A green area represents a hill or large open space. A small info box at the bottom right shows 'Settat Province Maroc' with coordinates '33.031024, -7.616685'. A legend at the bottom right includes icons for people, traffic, and zoom controls.

```
import UIKit
import MapKit

class ViewController: UIViewController, MKMapViewDelegate {

    @IBOutlet weak var myMap: MKMapView!

    override func viewDidLoad() {
        super.viewDidLoad()

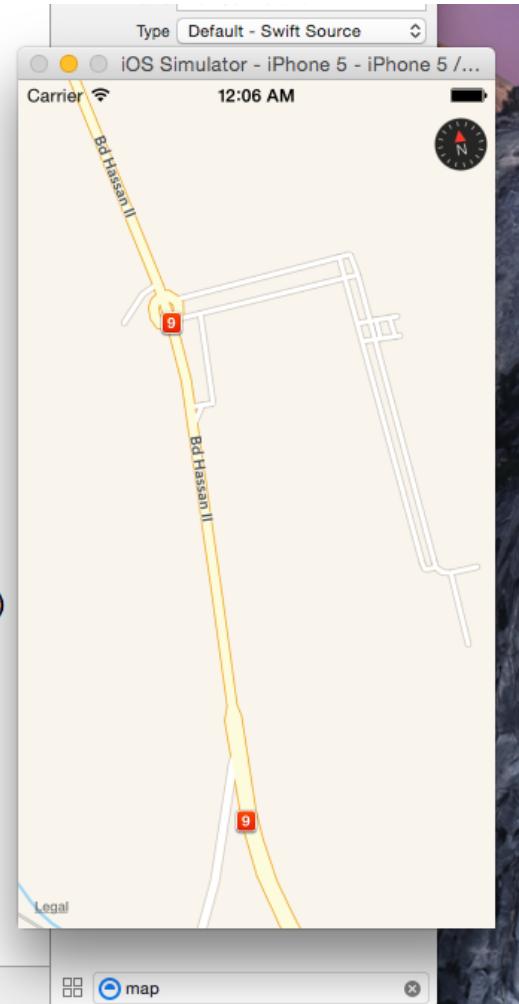
        var latitude:CLLocationDegrees = 33.030980
        var longitude : CLLocationDegrees = -7.616516

        var latDelta : CLLocationDegrees = 0.01
        var lonDelta : CLLocationDegrees = 0.01

        var span : MKCoordinateSpan = MKCoordinateSpanMake(lonDelta, lonDelta)
        var location : CLLocationCoordinate2D = CLLocationCoordinate2DMake(latitude, longitude)
        var region : MKCoordinateRegion = MKCoordinateRegionMake(location, span)

        myMap.setRegion(region, animated: true)
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```



Youssef  
BALOUKI

# Ajout des annotations aux Maps

```
override func viewDidLoad() {
    super.viewDidLoad()

    var latitude:CLLocationDegrees = 33.030980
    var longitude : CLLocationDegrees = -7.616516

    var latDelta : CLLocationDegrees = 0.01
    var lonDelta : CLLocationDegrees = 0.01

    var span : MKCoordinateSpan = MKCoordinateSpanMake(lonDelta , lonDelta)
    var location : CLLocationCoordinate2D = CLLocationCoordinate2DMake(latitude,
        longitude)
    var region : MKCoordinateRegion = MKCoordinateRegionMake(location, span)

    myMap.setRegion(region, animated: true)

    var annotation = MKPointAnnotation()

    annotation.coordinate = location

    annotation.title = "Faculté des Sciences et Techniques FSTS"
    annotation.subtitle = "I go here...some time"

    myMap.addAnnotation(annotation)

}
```



# Ajout des annotations aux Maps

```
var annotation = MKPointAnnotation()
annotation.coordinate = location
annotation.title = "FSTS"
annotation.subtitle = "I go here...some time"
myMap.addAnnotation(annotation)

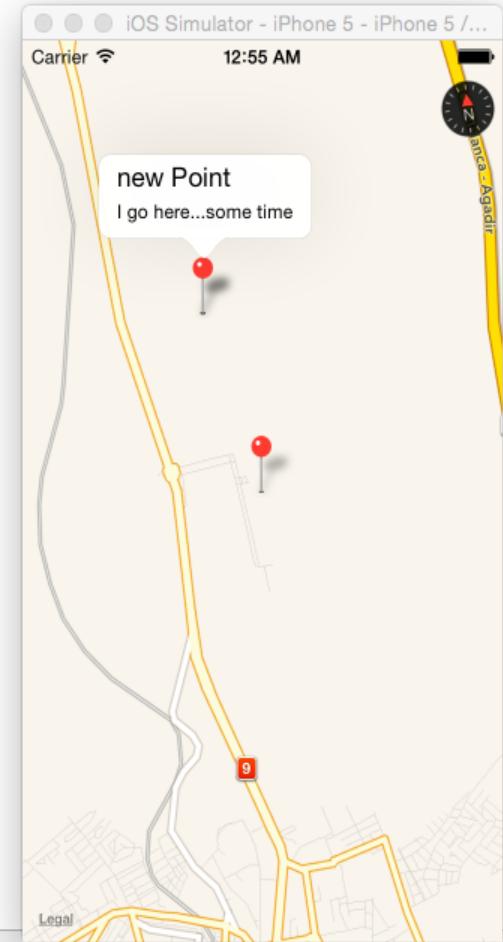
var uilpgr = UILongPressGestureRecognizer(target: self, action: "action:")
uilpgr.minimumPressDuration = 2.0
myMap.addGestureRecognizer(uilpgr)

}

func action(gesture : UIGestureRecognizer){

    print("test"  )
    var touchpoint = gesture.locationInView(self.myMap )
    var newcoordinate : CLLocationCoordinate2D = myMap.convertPoint(touchpoint,
    toCoordinateFromView: self.myMap)

    var newannotation = MKPointAnnotation()
    newannotation.coordinate = newcoordinate
    newannotation.title = "new Point"
    newannotation.subtitle = "I go here...some time"
    myMap.addAnnotation(newannotation)
```



# TP : Map + Géolocalisation

- Tracer le parcours réalisé par un cycliste en indiquant sa position courante par des annotation sur la carte.
  - 1: importer les deux framework : CoreLocation, MapKit.
  - 2: définir les propriétés de infoList
  - 3: les Labels nécessaires pour afficher latitude, longitude, altitude et distance
  - 4: ajouter un MapView
  - 5: écrire le code pour chaque actions
  - 6: établir le lien entre les vues de storyBoard et le code .

# TP : Map + Géolocalisation

The image shows a screenshot of the Xcode IDE. On the left, the storyboard editor displays a view controller interface with several UI elements: a MKMapView at the bottom, a button labeled "Button" above it, and seven labels arranged in two columns. The labels are: "Current latitude", "Current longitude", "Horizontal", "altitude", "Vertical", "Label", and "Label". The "Horizontal" label has a red border. On the right, the Xcode editor shows the corresponding Swift code:

```
import UIKit
import CoreLocation
import MapKit

class ViewController: UIViewController, CLLocationManagerDelegate {
    @IBOutlet weak var map: MKMapView!
    @IBOutlet weak var latitude: UILabel!
    @IBOutlet weak var dis: UILabel!
    @IBOutlet var label: UILabel!
    @IBOutlet weak var longitude: UILabel!
    @IBOutlet weak var vaccuracy: UILabel!
    @IBOutlet weak var altitude: UILabel!
    @IBOutlet weak var HAccuracy: UILabel!
    var locationManager: CLLocationManager = CLLocationManager()
    var startLocation: CLLocation!
    var startLocation1: CLLocation!
    var msg = MKPointAnnotation()

    @IBAction func reset(_ sender: Any) {
        startLocation = nil
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        locationManager.desiredAccuracy = kCLLocationAccuracyBest
        locationManager.delegate = self
        locationManager.requestWhenInUseAuthorization()
        locationManager.startUpdatingLocation()
        startLocation = nil
    }
}
```

The code implements a CLLocationManagerDelegate for the map view and handles location updates.

# TP : Map + Géolocalisation

The screenshot shows the Xcode interface with a storyboard on the left and a code editor on the right.

**Storyboard:** The storyboard contains a single view controller with the following components:

- A MKMapView at the bottom.
- Four labels on the top-left:
  - "Current latitude" with placeholder "Current"
  - "Current longitude" with placeholder "Current"
  - "Horizontal" with placeholder "HORIZ"
  - "altitude" with placeholder "ALTIT"
- Two labels on the top-right:
  - "Vertical" with placeholder "Vrtica"
  - "Label" (empty)
- A "Button" on the right side.

**ViewController.swift:** The code in the code editor is as follows:

```
func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {
    let latestLocation: CLLocation = locations[locations.count - 1]
    latitude.text = String(format: "%.4f", latestLocation.coordinate.latitude)
    longitude.text = String(format: "%.4f", latestLocation.coordinate.longitude)
    HAccuracy.text = String(format: "%.4f", latestLocation.horizontalAccuracy)
    altitude.text = String(format: "%.4f", latestLocation.altitude)
    vaccuracy.text = String(format: "%.4f", latestLocation.verticalAccuracy)

    let span : MKCoordinateSpan = MKCoordinateSpanMake(0.01, 0.01)
    let loc : CLLocationCoordinate2D = CLLocationCoordinate2DMake(latestLocation.coordinate.latitude, latestLocation.coordinate.longitude)
    let region : MKCoordinateRegion = MKCoordinateRegionMake(loc, span)

    CLGeocoder().reverseGeocodeLocation(latestLocation) {(mar,err) in
        if let place = mar?[0] { self.label.text = place.addressDictionary?.description }

        msg.coordinate = latestLocation.coordinate
        msg.title = "je suis ici"
        msg.subtitle = "Mustaph NAJARI"
        map.addAnnotation(msg)
        map.setRegion(region, animated: true)
        map.showsUserLocation = true

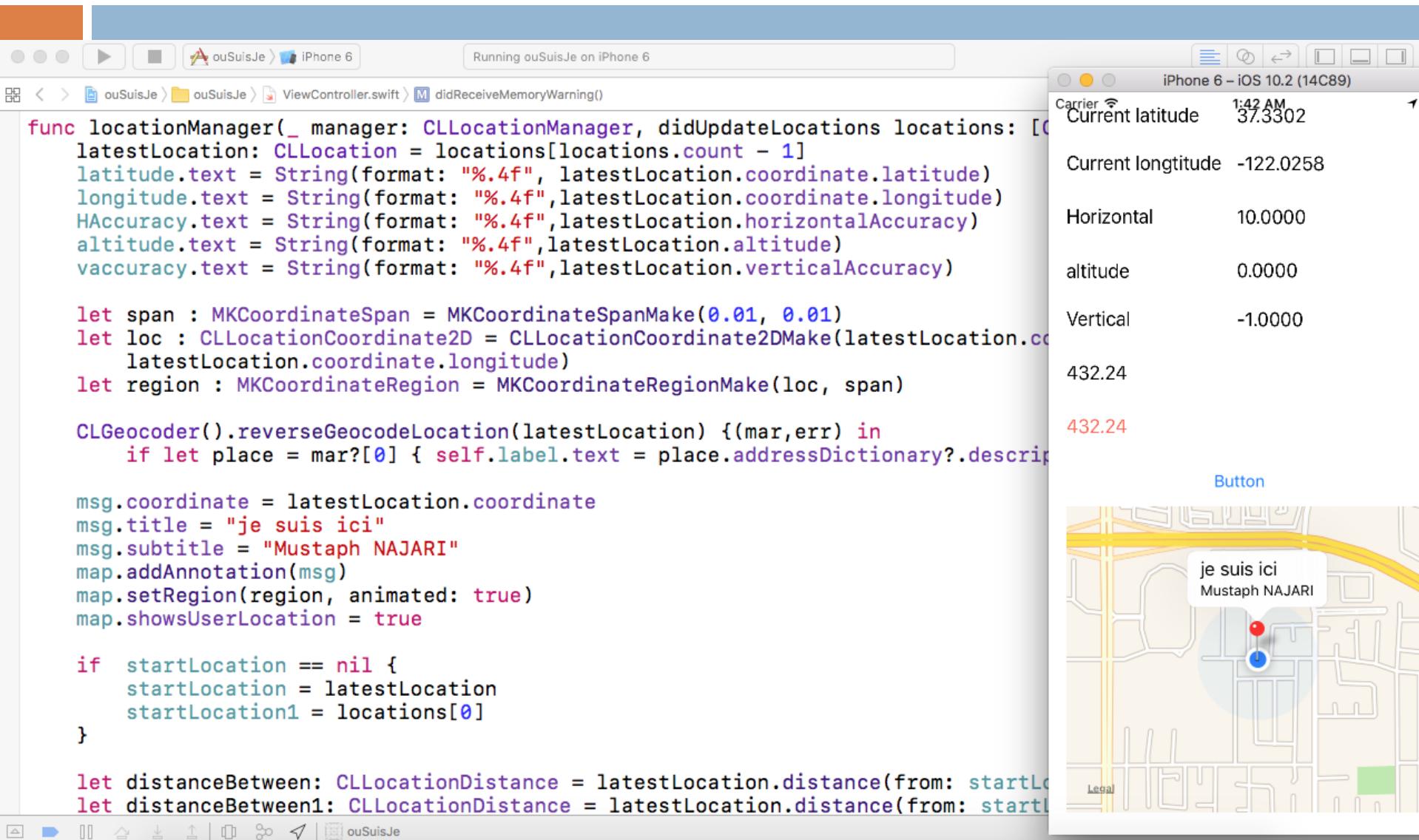
        if startLocation == nil {
            startLocation = latestLocation
            startLocation1 = locations[0]
        }

        let distanceBetween: CLLocationDistance = latestLocation.distance(from: startLocation)
        let distanceBetween1: CLLocationDistance = latestLocation.distance(from: startLocation)

        label.text = String(format: "%.2f", distanceBetween)
        dis.text = String(format: "%.2f", distanceBetween1)
    }
}
```

The code handles location updates and performs a reverse geocoding request to get the address for the latest location. It also sets up annotations on the map and calculates distances between the latest location and the first location in the array.

# TP : Map + Géolocalisation



The screenshot shows the Xcode IDE with a Swift file named ViewController.swift open. The code implements location updates and reverse geocoding to display the user's current location on a map.

```
func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {
    latestLocation = locations[locations.count - 1]
    latitude.text = String(format: "%.4f", latestLocation.coordinate.latitude)
    longitude.text = String(format: "%.4f", latestLocation.coordinate.longitude)
    HAccuracy.text = String(format: "%.4f", latestLocation.horizontalAccuracy)
    altitude.text = String(format: "%.4f", latestLocation.altitude)
    vaccuracy.text = String(format: "%.4f", latestLocation.verticalAccuracy)

    let span : MKCoordinateSpan = MKCoordinateSpanMake(0.01, 0.01)
    let loc : CLLocationCoordinate2D = CLLocationCoordinate2DMake(latestLocation.coordinate.latitude, latestLocation.coordinate.longitude)
    let region : MKCoordinateRegion = MKCoordinateRegionMake(loc, span)

    CLGeocoder().reverseGeocodeLocation(latestLocation) {(mar,err) in
        if let place = mar?[0] { self.label.text = place.addressDictionary?.description }

        msg.coordinate = latestLocation.coordinate
        msg.title = "je suis ici"
        msg.subtitle = "Mustaph NAJARI"
        map.addAnnotation(msg)
        map.setRegion(region, animated: true)
        map.showsUserLocation = true

        if startLocation == nil {
            startLocation = latestLocation
            startLocation1 = locations[0]
        }

        let distanceBetween: CLLocationDistance = latestLocation.distance(from: startLocation)
        let distanceBetween1: CLLocationDistance = latestLocation.distance(from: startLocation1)
    }
}
```

The right side of the interface shows the iPhone 6 simulator running iOS 10.2. The console output displays the current location coordinates and accuracy:

Carrier	Current latitude	37.3302
	Current longitude	-122.0258
Horizontal	10.0000	
altitude	0.0000	
Vertical	-1.0000	
432.24		
	432.24	

The simulator screen shows a map with a red dot at the user's location and a callout bubble containing the text "je suis ici" and "Mustaph NAJARI". A blue dot represents the user's current location.