



PROGRAMMATION IOS

<http://bit.ly/2h79B9t>

2016-2017 - LP

Cours/TD

Introduction générale

Quelques éléments d'Objective-C (Cours + Exercices)

Patterns sous Cocoa :

- structuration d'une application MVC , pattern cible-action
- datasource, delegate, notification

Tableau, Dictionnaire, Gestion de la mémoire (manuel, libération automatique)

Notification Push (certificat SSL, bi-clé RSA privé/publique – signature)

Quelques éléments du langage Swift



PROGRAMMATION IOS

TP "live"

Prise en main d'XCODE et Interface Builder

Hello, carte, Gestion des objets de l'interface, patterns avancés,
géolocalisation

Prise en main de Swift

*Ce que vous
devez (s)avoir
avant de
commencer ...*



*Premiers TP:
Hello, carte,
Localisation*

Objective-C

Swift
Intégration Swift
Et Objective-C



- Le marché du mobile : vers une diminution du nombre de SE mobile
- Les (nouveaux) eldorados :
 - L'internet des objets , les objets connectés, les objets wearables: ont pour but d' étendre Internet au monde réel en associant des étiquettes munies de codes, de puces RFID ou d'URL aux objets et aux lieux (réalité augmentée, d'expérience utilisateur,...)
 - la recherche et le partage d'information , jouer et communiquer : big data, data mining, web intelligent, intelligence artificielle, deep learning
- Les moyens associés aux smartphones
 - Capteurs et smartphone
 - Cloud computing :
 - un back-end pour stocker les données relatives à vos applications, - des web services ou microservices de type REST pour les rendre accessibles sous la forme de requête CRUD via les méthodes http (POST, GET, DELETE, PATCH).
 - <http://www.zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.htm>
 - <http://www.lesnumeriques.com/mobilite/parts-marche-os-mobiles-ios-stabilise-en-attendant-iphone-7-n55411.html>

Un retour vers le passé

Le mobile a déjà 30 ans et est devenu au fil des années un canal incontournable. Voici quelques terminaux qui ont marqué l'histoire :



Motorola
DynaTAC
8000x
(1983)



Bi-Bop
(1990)



Nokia 3210
et 3310
(1999)



Blackberry
5790
(2000)



Motorola
Razr V3
(2005)



Apple
iPhone 1
(2007)

Ce canal sera bientôt dominant, en nombre de connexion dans le monde...

Dès maintenant “[...] 4 consommateurs sur 5 peuvent accéder à Internet et au commerce en ligne via leur mobile, contre 1 consommateur sur 5 via son ordinateur.”

(source: TNS Sofres) 2012

Dans le monde, en 2014, Internet sera majoritairement mobile en nombre de connexion
(Source: Etude Morgan Stanley 2011)

Un retour vers le passé

Partout et tout le temps

- > 90% des utilisateurs gardent leur smartphone sur eux 24h/24h
- Majoritairement utilisé à la maison...
 - Devant la télévision
 - Au lit
 - Dans la salle de bain ou aux toilettes
- ... et ensuite en « mobilité » :
 - Durant une attente
 - En transport
 - En réunion / en cours
 - En faisant les courses
- L'âge moyen du premier mobile :
 - 11 ans et demi !

Portrait du Mobinaute

Données France

- Près de 50% ont moins de 35 ans
- 8/10 habitent hors agglomération parisienne (vs. 3/4 l'an dernier)
- 43% sont des femmes (vs. 40% l'an dernier)

(Source: Médiamétrie)

Source : IPSOS et CETELEM



Un retour vers le passé

Terminaux, OS* et navigateurs : la problématique se déplace des terminaux vers les OS

*OS = *operating system i.e.*
système d'exploitation

- **Terminaux... du mieux !**

- La montée en puissance d'Android ainsi que l'accord Nokia/Microsoft ont permis de réduire le nombre d'OS majeurs présents sur le marché grand public
- Là où il fallait parfois viser une multitude de terminaux, on peut aujourd'hui par exemple considérer qu'une application développée pour Android 2.1 fonctionnera sur tout appareil qui tourne sous Android 2.1

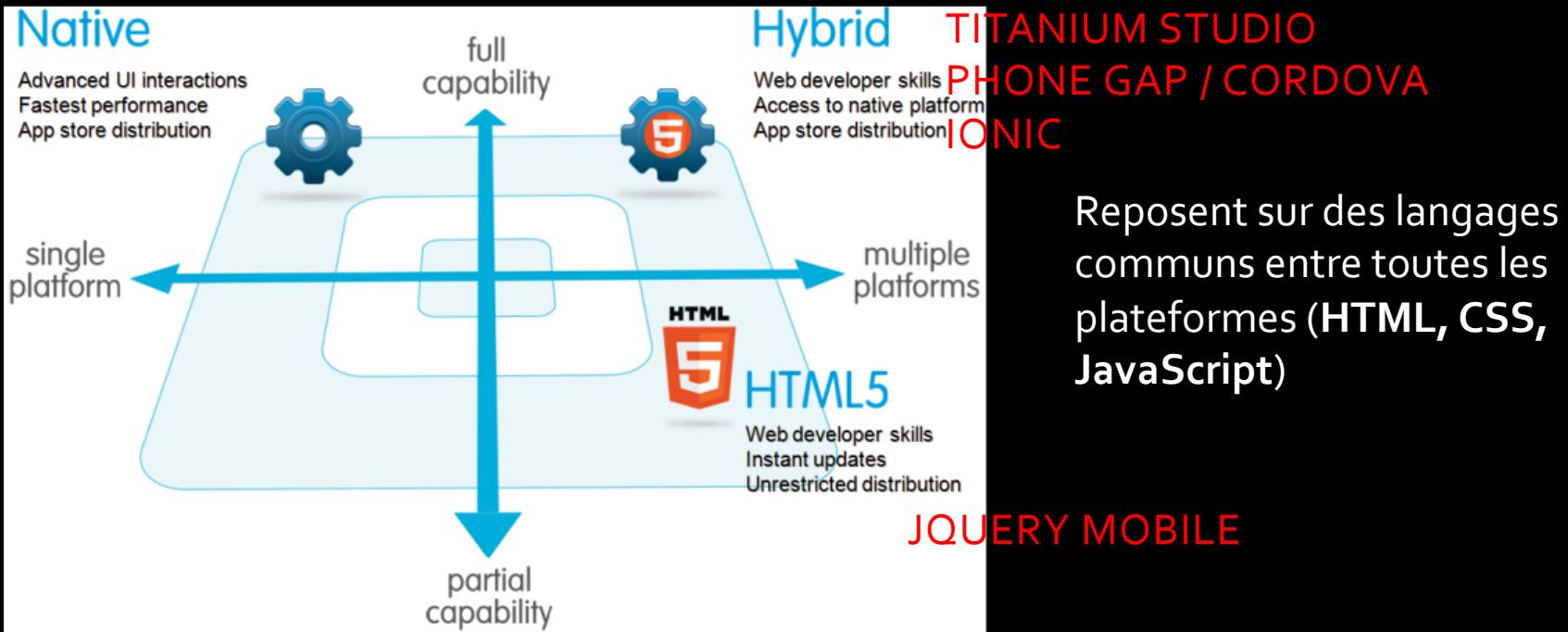
Problématique modèle
et marque



Problématique OS et
navigateur

- Pour autant les OS et navigateurs sont toujours trop hétérogènes

- DÉVELOPPEMENT D'APPLICATIONS MOBILES HYBRIDE



Environnements de développement sur lesquels vous pouvez développer pour iPhone, système Android, iPad, ou simplement des **applications de bureau**.

A partir d'une **web app**, intégration à un **framework**
- qui la transforme en applications natives (hybride)
- ou l'adapte à un moteur de rendu web (phonegap)

Cordova/ ex-PhoneGap



Transformer des applications HTML/
Javascript en applications
installables sur un mobile (pas
besoin de serveur)

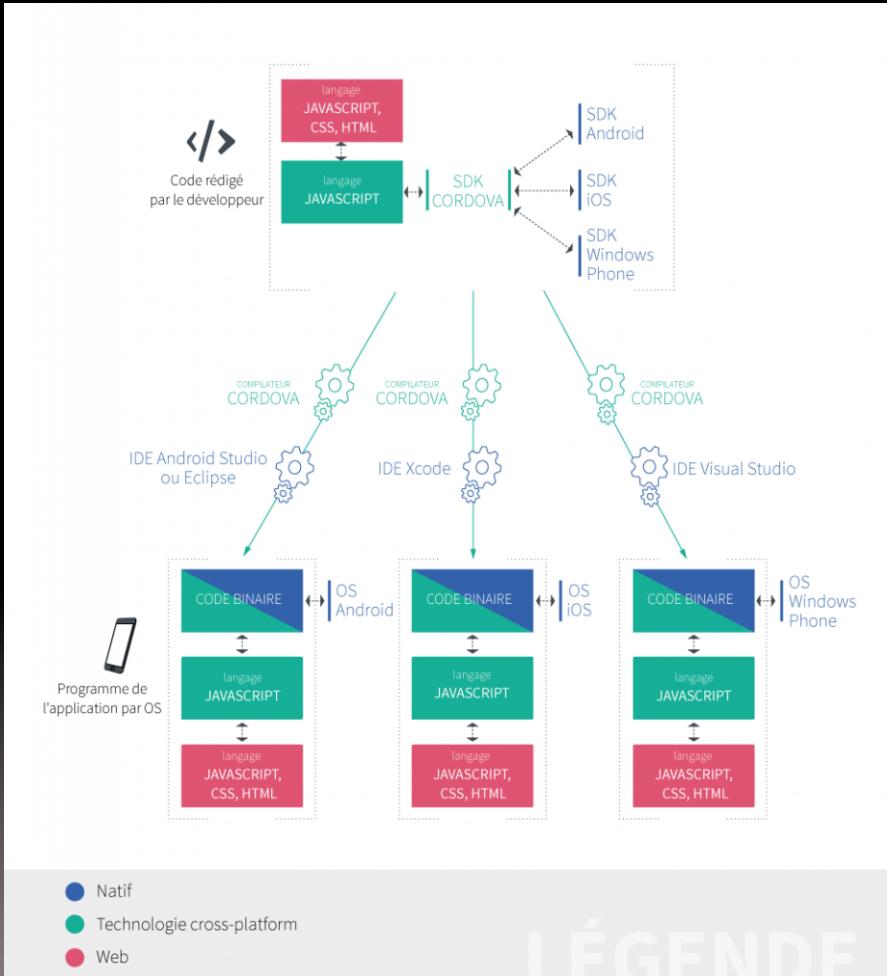
S'appuyer sur les technologies web :

- utiliser HTML5 pour la structure
- JavaScript pour accéder aux fonctionnalités Natives
- CSS3 pour le look and feel

Collection d'outils de cross
développement

et des API permettant l'accès aux
capteurs de chaque appareil

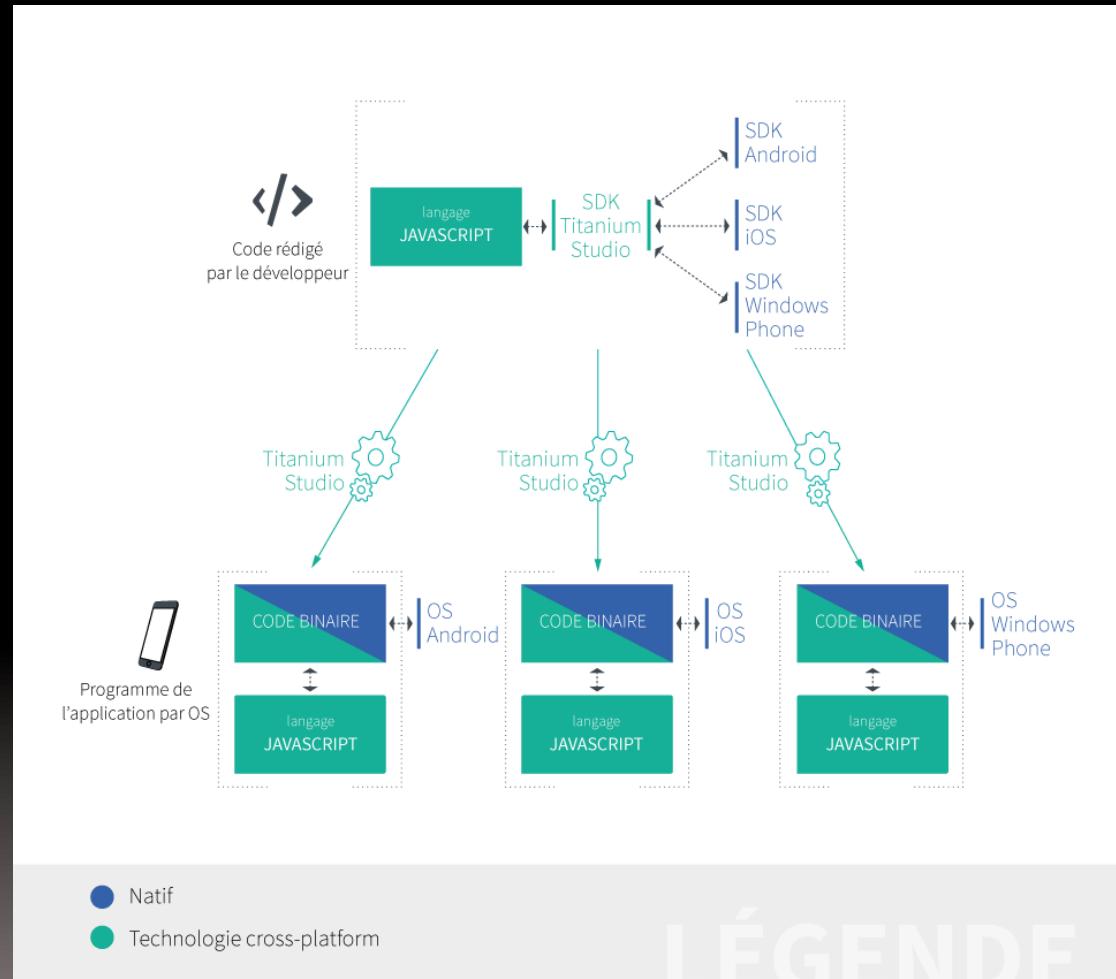
Les applications s'exécutent dans
des wrappers ciblés pour chaque
plateforme



Titanium



- Framework open source pour construire des applications multiplateformes « natives » pour mobile ou PC en s'appuyant sur Javascript
- Bibliothèques natives qui font le lien entre les appels Javascript et les éléments d'interface natifs, le stockage de données, la géolocalisation etc.



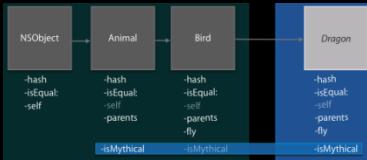
Développement NATIF : iPhone – IPAD



Xcode, IB, Cocoa Touch,
Frameworks



Objective-c et Swift



Accéléromètre



Appareil photo et camera



Interfaces



Sites web



Audio





Quelques sites à connaître

<http://blog.octo.com/tag/iphone/>

[https://openclassrooms.com/courses/programmez-en-objective-c/
introduction-a-l-objective-c](https://openclassrooms.com/courses/programmez-en-objective-c/introduction-a-l-objective-c)

<http://www.ipup.fr/index.php?/forum/8-tutoriels-programmer-sur-iphone/>

<https://developer.apple.com/develop/>

<https://developer.apple.com/>

<https://developer.apple.com/wwdc/videos/>



Watch Session Videos ▶

- Quelques exemples de video
 - Réalité augmentée sur l'iphone



Paris en réalité augmentée
Metro Paris



Iphone dans l'iphone. Vidéo



Pocket Universe



Wikitude AR Travel Guide

[http://www.youtube.com/watch?
feature=player_embedded&v=XRTLk13FXkk](http://www.youtube.com/watch?feature=player_embedded&v=XRTLk13FXkk)



Theodolite



Augmented driving

<http://www.clubic.com/article-324608-1-realite-augmentee-applications-iphone-3g.html>
http://cyberloire.myurm.biz/xwiki/bin/view/NBFAQ/Ressources_JKED7OYeSK2D



iOS 9

Explore the APIs that your apps can use in iOS 9.

[iOS Library](#)



OS X

Learn about developing for OS X El Capitan.

[OS X Library](#)



watchOS 2

Find out how to create great apps for Apple Watch.

[watchOS Library](#)



tvOS

Discover tvOS, the platform for the new Apple TV.

[tvOS Library](#)



Safari

Build web-based sites and apps with web standards HTML, CSS, and Javascript.

[Safari Library](#)



iAd

Create compelling rich media advertisements for the iAd App Network.

[iAd Library](#)



Legacy

Find documentation for developing with older versions of iOS and OS X.

[Retired Documents](#)

Apple Watch depuis le 24 avril
Apple Music depuis le 30 juin
iOS9.2 (SIRI, Notes, multi-fenêtre)
iPhone 6S , 6S+, 6C (4 pouce, Rétina)
iPadPro A9X, 12,9 pouce

Watch OS2 (gestion de capteurs de santé)
TV OS 9.1 pour l'Apple TV4
XCODE 7.1
OSX 10.11.12 El Captain
3D Touch, nouvelle puce (70% plus rapide que A8)
+ 4000 API cette année

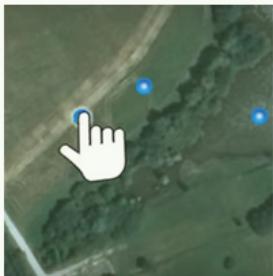
Projet – 3 semaines



Représentation de la
trajectoire à partir des données
NMEA + Information vitesse



Vue 1



Définition d'une trajectoire à
partir de waypoints.

Définition de la vitesse sur
chaque waypoint

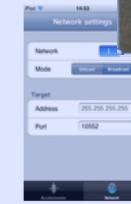
Transmission des données trajectoires
sous la forme d'un fichier json ou de trames NMEA

Vue 3



Vue 2

accel.x
accel.y
accel.z



iPod

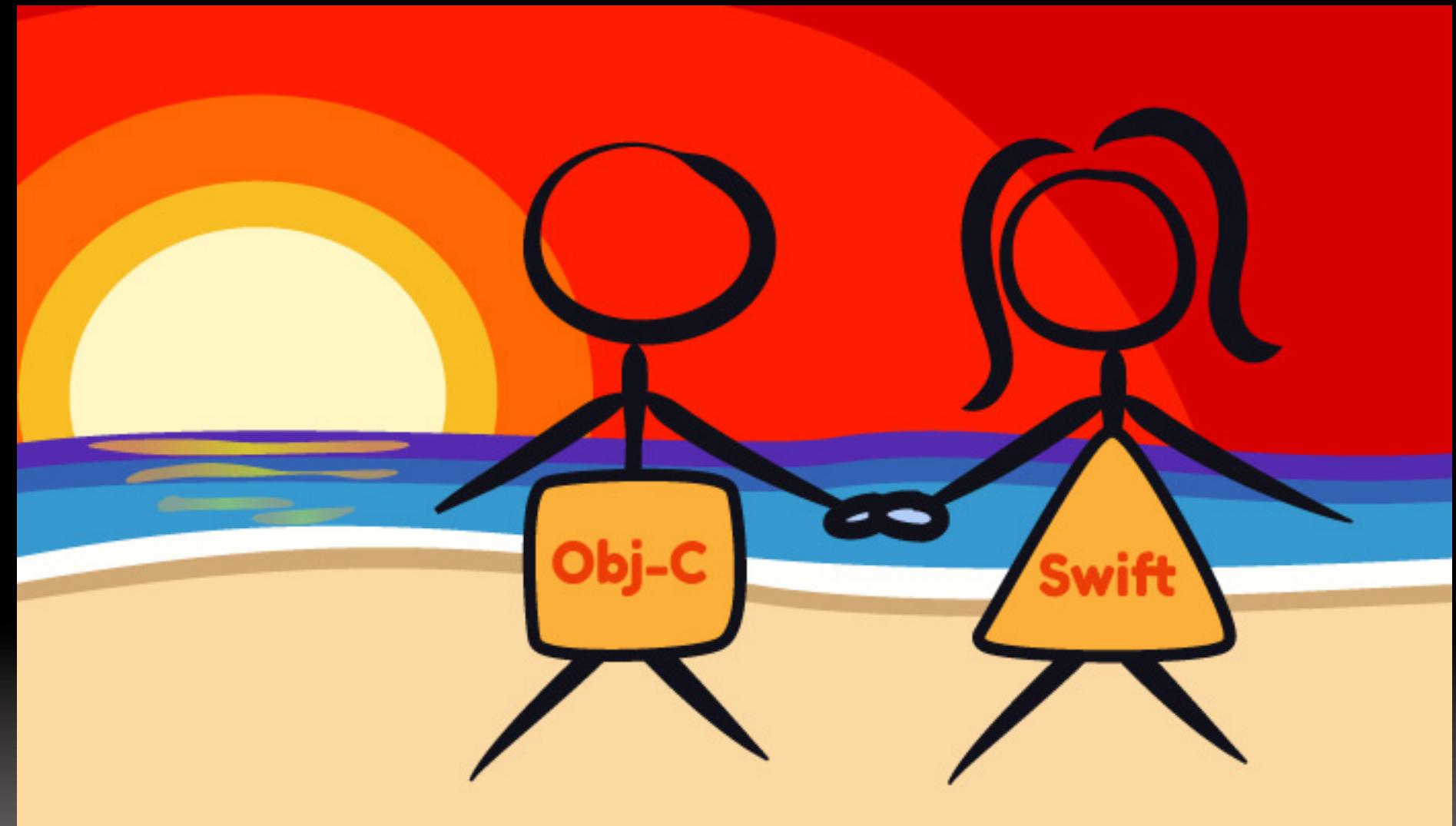


Simulation du déplacement d'un drone virtuel
à partir des données de l'accéléromètre



RouteToGPX





*Ce que vous
devez (s)avoir
avant de
commencer ...*

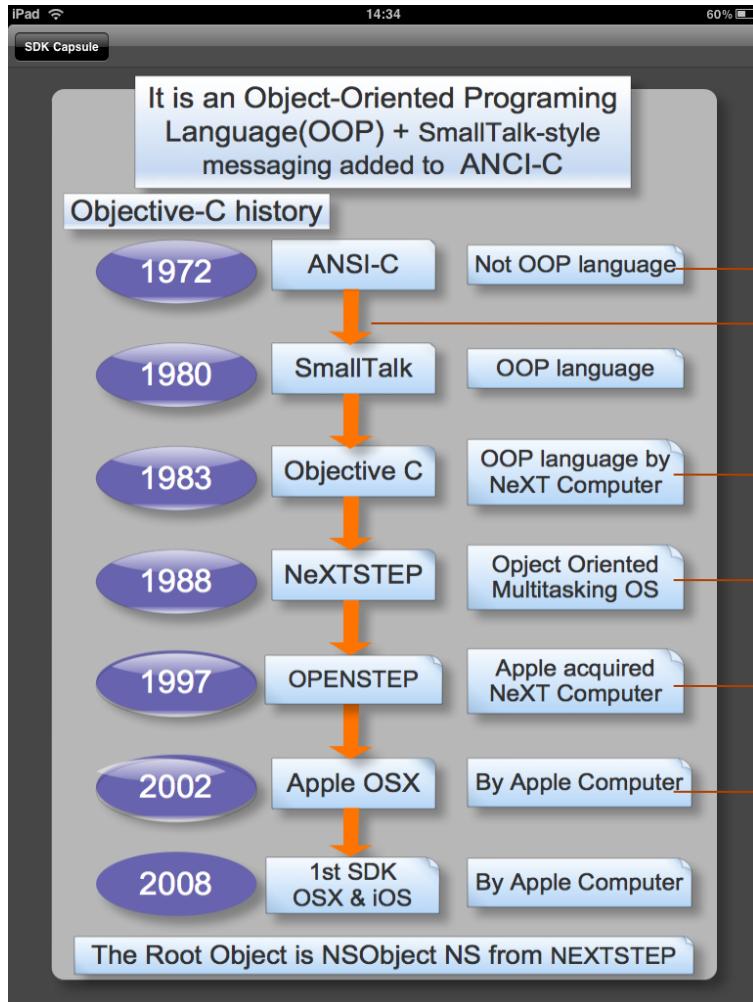
Objective-C



*Premiers TP:
Hello, carte,
Localisation*

*Swift:
Intégration Swift
Et Objective-C*

Quelques mots sur objective-C...



70 : UNIX et langage C (Ken Thompson, Dennis Ritchie)

79 : Bjarn Stroustrup développe le C++ ajoute le POO au C

Brad Cox



Entreprise NeXT créée par Steve Jobs (suite à son départ d'Apple). Rachetée en 1996 par Apple.

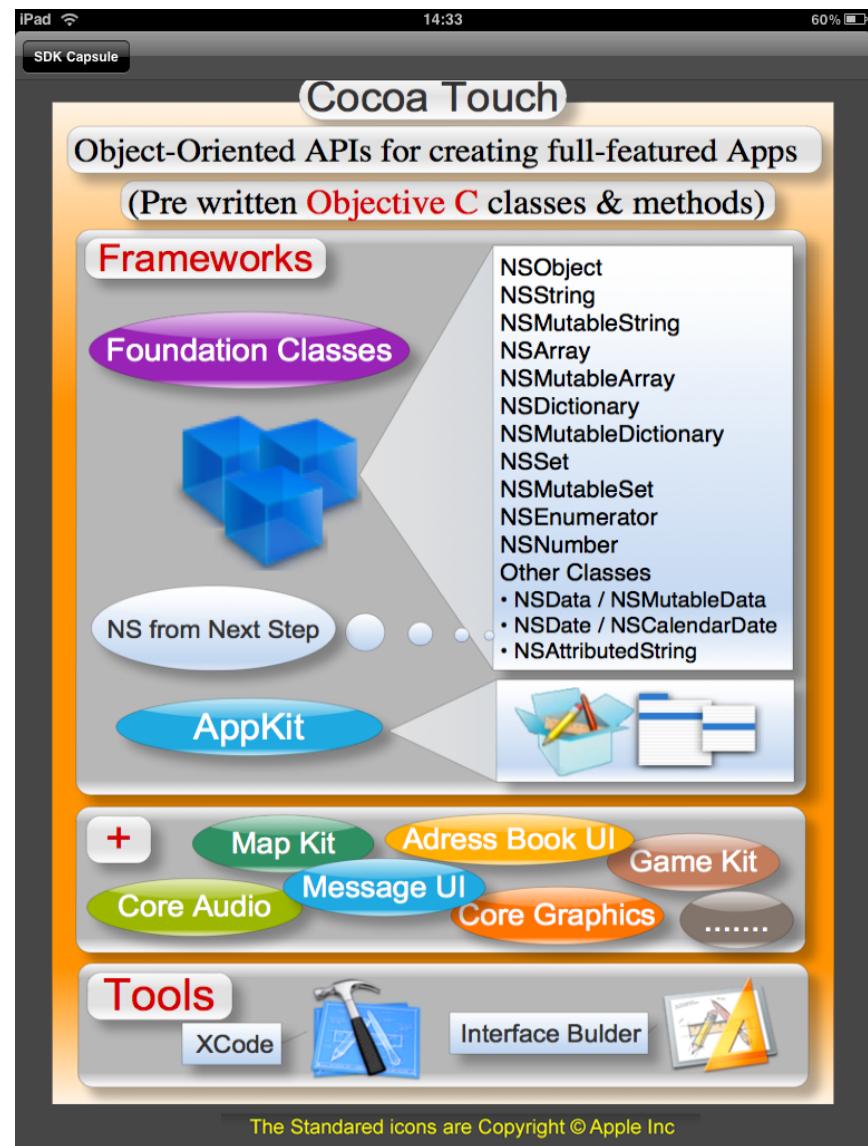
API libre

Basé sur BSD : propriétaire.
Noyau et structure interne du système libre
API Cocoa basé sur le framework de NeXTStep
(donc écrite en objective-c)

CocoaTouch et Frameworks

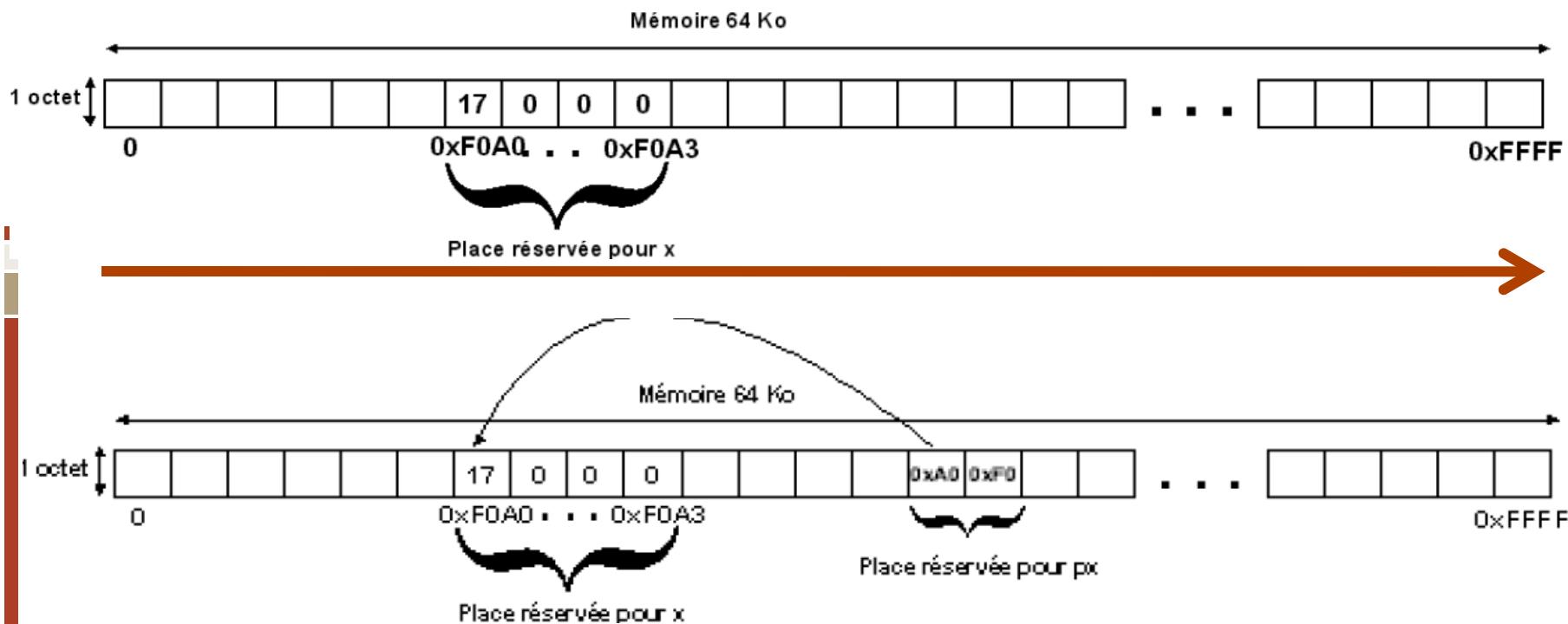
<https://developer.apple.com/reference/>

<http://developer.apple.com/library/ios/navigation/>



Rappel – Les pointeurs

- Par son adresse (adressage indirect)
 - une variable de type pointeur contient l'adresse d'une autre variable
 - Le lien entre pointeur et la variable pointée est géré par le programmeur
 - exemple:
 - `int x = 17;`
 - `int * px = &x;`

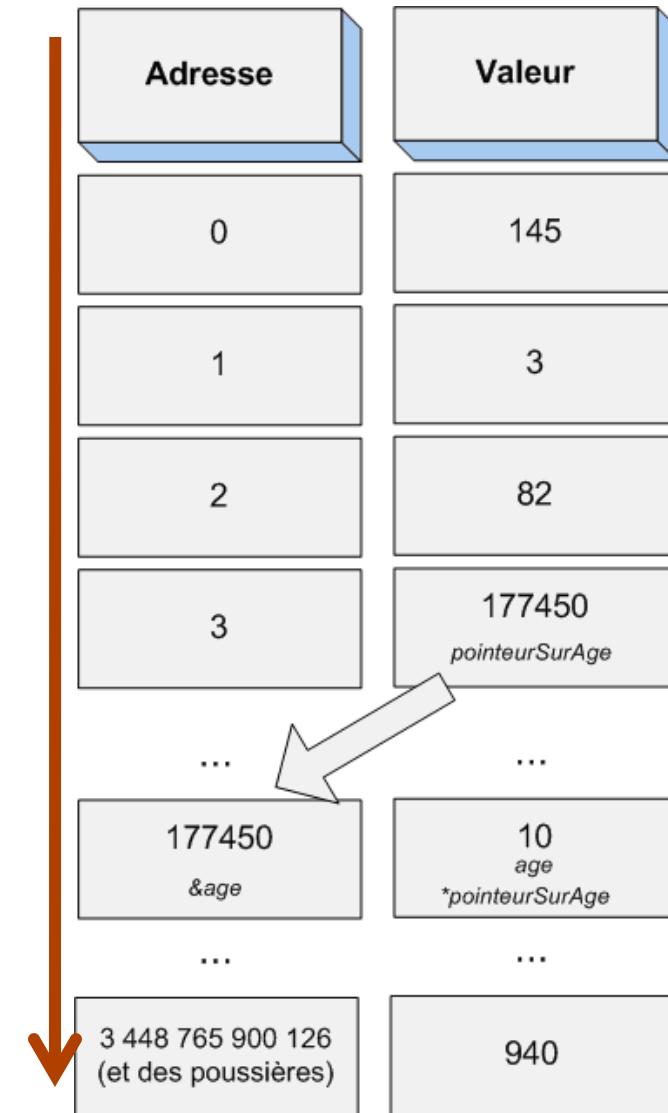
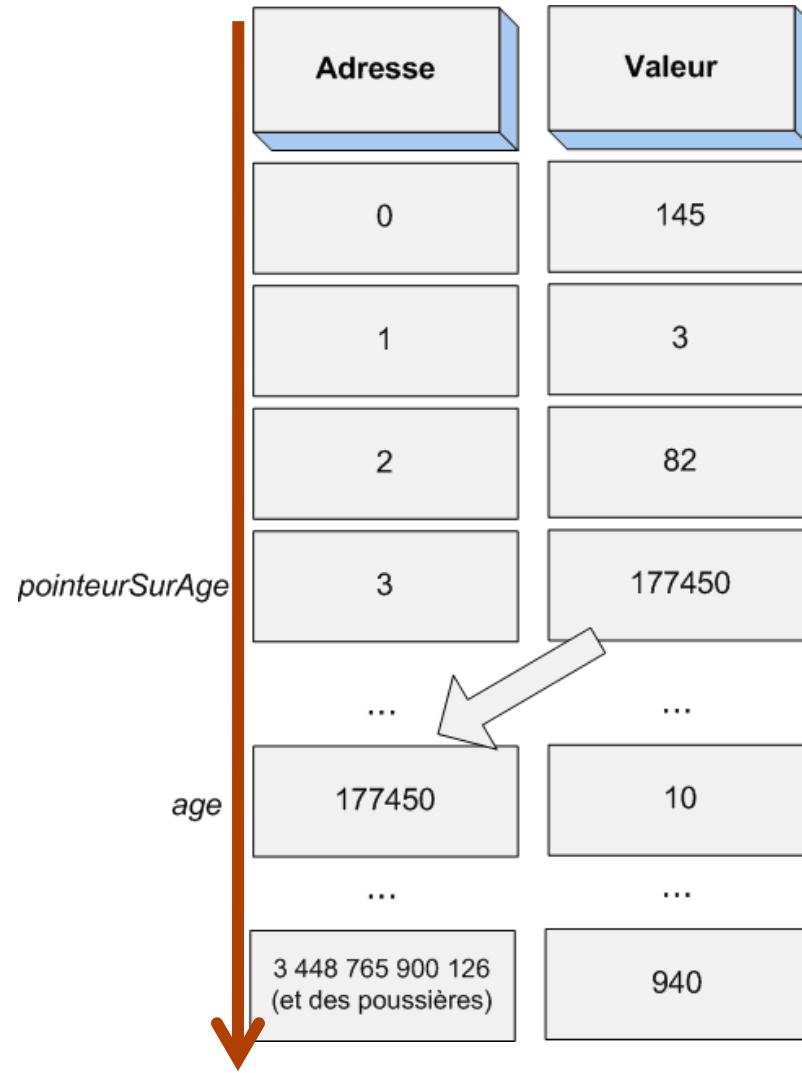


Rappel - Les pointeurs

```
int age = 10;
```

```
int *pointeurSurAge;
```

```
pointeurSurAge = &age;
```





|| Quelques mots sur objective-C...

Les grands principes

Classe NSObject

Tout est objet

Tous les objets héritent de la classe NSObject

Toutes les méthodes de NSObject sont donc accessibles dans tous les objets créés.

Ex : listes et dictionnaires peuvent être exploités avec tout type d'objet.

runtime

Envoie de messages

Les objets communiquent entre eux grâce à des envois de messages qui sont gérés par le runtime.

A l'exécution, le message est transmis à l'objet destinataire.

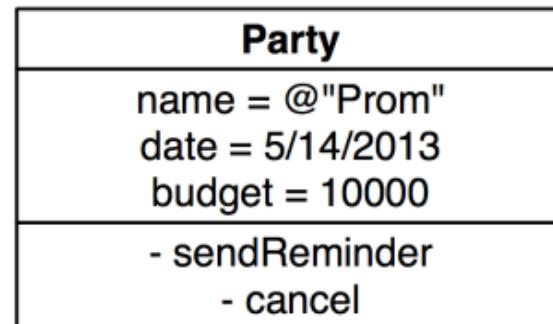
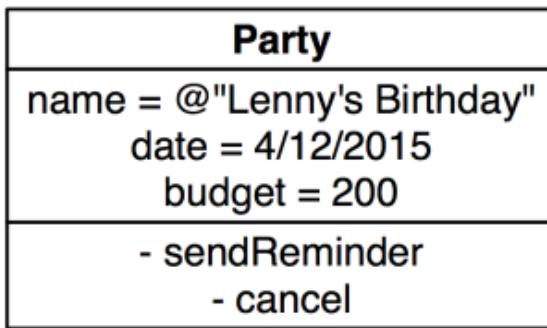
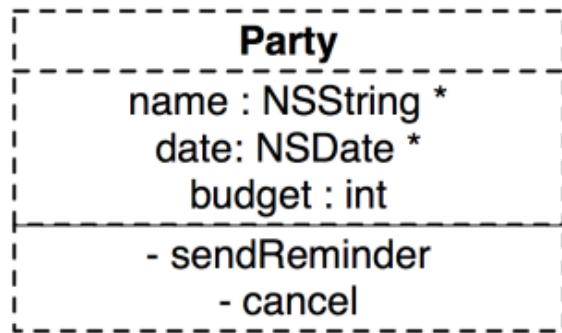
Si celui-ci ne sait pas y répondre une exception est levée. (*le compilateur n'a pas besoin de vérifier à la compilation que l'objet cible est capable de traiter ce message*).

http://developer.apple.com/library/mac/#documentation/Cocoa/Reference/Foundation/Classes/nsobject_Class/Reference/Reference.html



|| Quelques mots sur objective-C...

Une classe et deux instances de la classe





||| Quelques mots sur objective-C...

Définition de l'Objective-C (Wikipedia)

L'**Objective-C** est un langage de programmation orienté objet réflexif. C'est une extension du C ANSI, comme le C++, mais qui se distingue de ce dernier par sa distribution dynamique des messages, son faible typage, son typage dynamique et son chargement dynamique.

Réflexif : capacité à se modifier à l'exécution (ex : le type d'une variable peut changer à l'exécution).

Extension du C : hérite de la syntaxe du C : aucune incompatibilité entre le C et Objective –C

Typage dynamique et faible : le type d'une variable est défini à l'exécution.

Système de chargement dynamique : l'objective-C s'exécute dans un runtime, à la manière de Java : machine virtuelle qui se charge de distribuer les messages, de créer les objets et les classes ...
Inutile d'installer une machine comme Java.

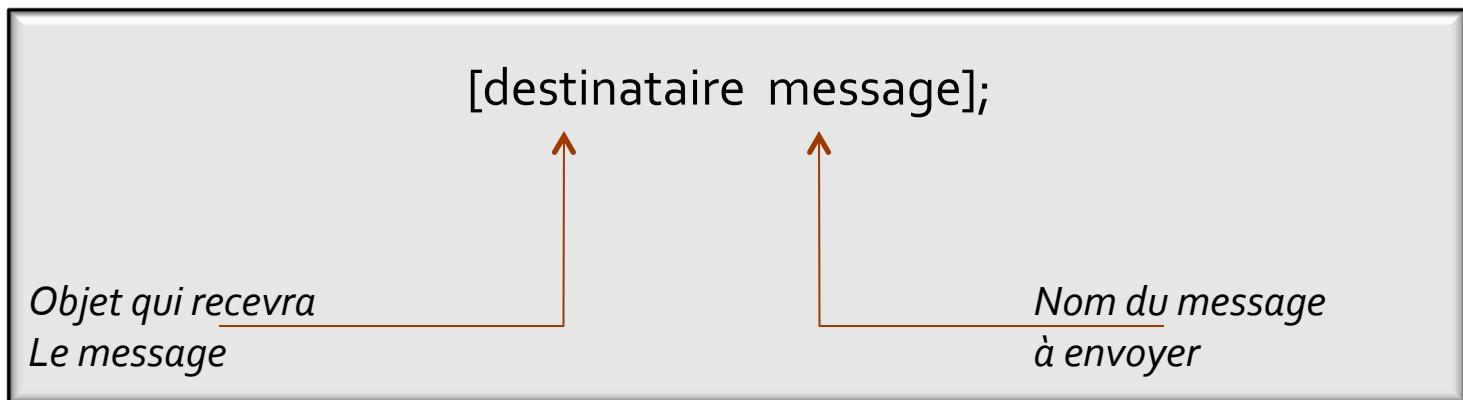


|| Quelques mots sur objective-C...

Syntaxe Objective-C

Toute la syntaxe C est reconnue et peut-être utilisée
Extensions visent à permettre la programmation orientée objet.

Envoyer des messages à un objet



Signature de la méthode
message

Equivalent en JAVA

destinataire.message();



|| Quelques mots sur objective-C...

Syntaxe Objective-C

Système de message – runtime – développement dynamique

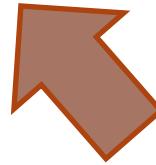
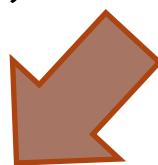
Interfaçage entre langages plus souple

Sérialisation de données

[B message];

Message Dispatcher

trying



Objet B

Objet A

<objc/message.h>
objc_msgSend(id Receiver, SEL
the selector,...);

L'objet qui envoie le message ne peut s'assurer que celui-ci arrivera à destination
→ compilation/exception à l'exécution



||| Quelques mots sur objective-C...

Syntaxe Objective-C

Système de message – runtime – développement dynamique

Interfaçage entre langages plus souple

Sérialisation de données

Respecte totalement le Key-Value Coding (KVC) : un objet quelconque peut être traité comme un dictionnaire.

```
monArticle.title =@"Un Article Super !";
[monArticle setValue:@"Un Article Super !" forKey:@"title"];
```

Si la propriété "title" n'existe pas, nous pouvons la rajouter directement depuis l'exécution du programme (et non pas avant la compilation) et accéder à cette propriété



||| Quelques mots sur objective-C...

Syntaxe Objective-C

Système de message – runtime – développement dynamique
Interfaçage entre langages plus souple
Sérialisation de données

Respecte totalement le Key-Value Coding (KVC) : un objet quelconque peut être traité comme un dictionnaire.

```
// On crée un pointeur vers un objet de la classe Personne, qu'on a d'abord alloué dans l'espace mémoire, et initialisé.  
Personne *exemple = [[Personne alloc] init];  
//On envoie un message au pointeur exemple qui va le rediriger vers l'objet vers lequel il pointe, pour fixer la valeur de la clef surname.  
[exemple setValue:@"Paul" forKey:@"surname"];
```

|| Quelques mots sur objective-C...



Syntaxe Objective-C

Envoyer des messages à un objet avec ajout de paramètres



Les paramètres des méthodes sont nommées (étiquette)
Le nom de la méthode représente aussi le nom du premier argument



|| Quelques mots sur objective-C...

Syntaxe Objective-C. *Le nommage des paramètres est une des spécificités du langage*

Exemple

```
[string stringByReplacingOccurrencesOfString:@ "iPhone " withString:@ "Ipod "];
```

Objet qui recevra
Le message

Nom du message
à envoyer

Les paramètres des méthodes sont nommés
Le nom de la méthode contient aussi le nom du label du second argument

Signature de la méthode : **stringByReplacingOccurrencesOfString: withString:**
Equivalent JAVA : string.replaceAll(' iphone' , "iPod");

Quelques mots sur objective-C...



Syntaxe Objective-C. *Le nommage des paramètres est une des spécificités du langage*

Autre exemple de message envoyé avec la méthode

`replaceObjectsInRange:withObjectsFromArray:range:`

Sélecteur @selector()

```
[arrayInstance replaceObjectsInRange:aRange  
    withObjectsFromArray:anotherArray  
    range:anotherRange];
```

Dans un autre langage :

La classe `NSString` a deux méthodes : `rangeOfString:options:`
`rangeOfString:options:range:`



|| Quelques mots sur objective-C...

Syntaxe Objective-C

Il est également possible de

-récupérer une valeur de retour

```
NSObject *returnedObject=[destinataire message:argument1];
```

-de chaîner plusieurs appels de méthodes

```
NSObject *returnedObject=[[destinataire message] faisQuelquechoseAvec];
```

Autre exemple déjà rencontré

```
NSString *str=[[NSString alloc] init];
```



Exercice 1

Soit la méthode déclarée de la manière suivante :

```
- (void) getCharacters:(unichar *) buffer range:(NSRange) aRange;
```

Quel est le nom de la méthode ?

S'agit-il d'une méthode d'instance ou de classe ?

Est-ce que la méthode retourne une valeur ?

Précisez les paramètres

Qu'est sensée faire cette méthode ?

Commentez le programme suivant :

```
int main(int argc, char *argv[])
```

1

```

NSString *myString=[[NSString alloc] initWithString:@"chaine de test"];
unichar *buffer=calloc ([myString length],sizeof(unichar));
[myString getCharacters: buffer      range:(NSMakeRange)(1,5)];
NSLog(@"test1");
printf("%c",buffer[3]);
free(buffer);

```

3

Soit l'exemple d'appel suivant d'une méthode sur un objet : [Paul eatWith:Pauline andSpeakAbout:Subject]. Construire la signature et la déclaration de la méthode appelée.

Autre méthode :

```
NSString *s2 =[s1 stringByReplacingOccurrencesOfString:@"[ab]"  
           withString:@"z"  
           options:NSRegularExpressionSearch  
           range:NSMakeRange(0, s1.length)];
```



||| Quelques mots sur objective-C...

Déclaration d'une classe

- L'interface décrit le comportement de la classe tel qu'il sera vu par les autres classes
- L'implémentation décrit la logique interne de la classe

Déclaration d'une classe: MaClasse.h
@interface MaClasse: NSObject {

// Liste des **variables d'instance** de la classe

NSString *monNom;
NSInteger *monAge;

}

// Liste des **méthodes** de la classe

- (void) uneMethodeDInstance;
+(void) uneMethodeDeClasse;

@end

Une classe hérite forcément de
NSObject ou d'un de ses descendants
L'héritage multiple n'est pas possible

Elles ne sont pas visibles
par les autres objets

par défaut, publique

Les variables d'instance sont l'équivalent des propriétés en java ou php. En objective C, une propriété est une notion différente.



|| Quelques mots sur objective-C...

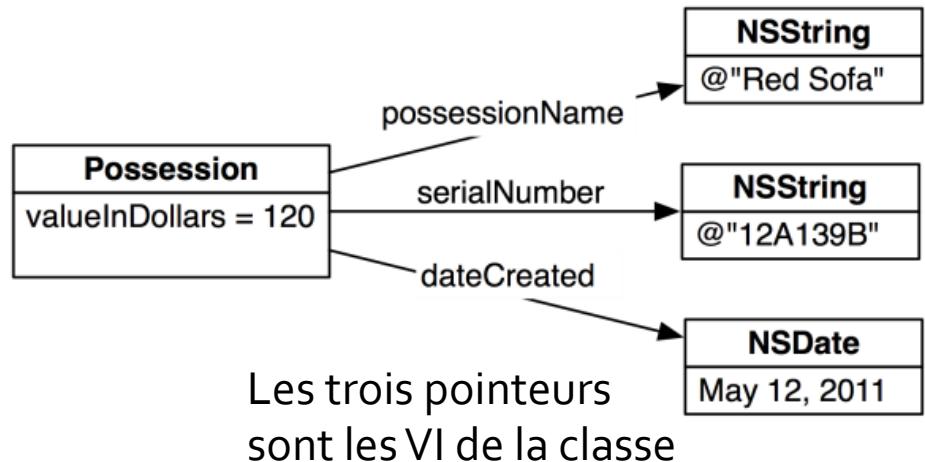
Déclaration d'une classe

Exemple

```
#import <Foundation/Foundation.h>

@interface Possession : NSObject
{
    NSString *possessionName;
    NSString *serialNumber;
    int valueInDollars;
    NSDate *dateCreated;
}
@end
```

Une instance de la classe Possession



||| Quelques mots sur objective-C...



Exemple

```
#import <Foundation/Foundation.h>

@interface Possession : NSObject
{
    NSString *possessionName;
    NSString *serialNumber;
    int valueInDollars;
    NSDate *dateCreated;
}
@end

// Create a new Possession instance
Possession *p = [[Possession alloc] init];

// Set possessionName to a new NSString
[p setPossessionName:@"Red Sofa"];

// Get the pointer of the Possession's possessionName
NSString *str = [p possessionName];

// Print that object
NSLog(@"%@", str); // This would print "Red Sofa"

// Getter
- (NSString *)possessionName
{
    // Return a pointer to the object this Possession calls its possessionName
    return possessionName;
}

// Setter
- (void)setPossessionName:(NSString *)newPossessionName
{
    // Change the instance variable to point at another string,
    // this Possession will now call this new string its possessionName
    possessionName = newPossessionName;
}
```



||| Quelques mots sur objective-C...

Exemple

```
#import <Foundation/Foundation.h>

@interface Possession : NSObject
{
    NSString *possessionName;
    NSString *serialNumber;
    int valueInDollars;
    NSDate *dateCreated;
}
- (void)setPossessionName:(NSString *)str;
- (NSString *)possessionName;

- (void)setSerialNumber:(NSString *)str;
- (NSString *)serialNumber;

- (void)setValueInDollars:(int)i;
- (int)valueInDollars;

- (NSDate *)dateCreated;
@end
```

|| Quelques mots sur objective-C...



Implementation d'une classe

```
#import "MaClasse.h"

@implementation MaClasse
-(void) uneMethodeDInstance
{
//...
}

+(void) uneMethodeDeClasse
{
//...
}

@end
```



Ne peuvent être appelée que sur une instance de la classe



Appelées directement sur la classe. Elles n'ont pas accès aux variables d'instance de la classe

||| Quelques mots sur objective-C...



Implementation d'une classe

```
#import "MaClasse.h"

@implementation MaClasse
-(void) uneMethodeDInstance
{
//...
}

+(void) uneMethodeDeClasse
{
//...
}

@end
```

```
MaClasse *instance=[[MaClasse alloc] init];
[instance uneMethodeDInstance];
```

```
[MaClasse uneMethodeDeClasse];
```

```
MaClasse *instanceNonInitialisee=[MaClasse alloc];
MaClasse *instance=[instanceNonInitialisee init];
```

||| Quelques mots sur objective-C...

Exemple



```
#import "Possession.h"

@implementation Possession

- (void)setPossessionName:(NSString *)str
{
    possessionName = str;
}
- (NSString *)possessionName
{
    return possessionName;
}
- (void)setSerialNumber:(NSString *)str
{
    serialNumber = str;
}

- (NSString *)serialNumber
{
    return serialNumber;
}
- (void)setValueInDollars:(int)i
{
    valueInDollars = i;
}
- (int)valueInDollars
{
    return valueInDollars;
}
- (NSDate *)dateCreated
{
    return dateCreated;
}

Possession *p = [[Possession alloc] init];
// This creates a new NSString, "Red Sofa", and gives it to the Possession
[p setPossessionName:@"Red Sofa"];

// This creates a new NSString, "A1B2C", and gives it to the Possession
[p setSerialNumber:@"A1B2C"];

// We send the value 100 to be used as the valueInDollars of this Possession
[p setValueInDollars:100];
NSLog(@"%@", '%@ %@ %@ %d', [p possessionName], [p dateCreated],
       [p serialNumber], [p valueInDollars]);
```



|| Quelques mots sur objective-C...

Exemple

Méthode d'instance particulière (surcharge de la méthode description)

```
- (NSString *)description
{
    NSString *descriptionString =
        [[NSString alloc] initWithFormat:@"%@ (%@): Worth $%d, recorded on %@",
         possessionName,
         serialNumber,
         valueInDollars,
         dateCreated];

    return descriptionString;
}
```

Appel de la méthode d'instance

NSLog(@"%@", p);

→ Red Sofa (A1B2C): Worth \$100, recorded on (null)

Quelques mots sur objective-C...



Héritage

```
@interface MaDeuxiemeClasse:MaClasse{  
//...  
@end
```



Classe parente
(NSObject ou une de ses descendantes)

Une classe hérite toujours d'une seule classe

La notion de délégué de contrôle permet d'étendre le comportement d'une classe sans avoir à utiliser la notion d'héritage multiple.





|| Quelques mots sur objective-C...

Propriétés

La classe doit déclarer des accesseurs pour les variables d'instance qu'elle veut partager
Une propriété est une variable d'instance pour laquelle **des accesseurs** sont définis

```
@interface MaClasse: NSObject {  
    // Liste des variables d'instance de la classe  
    NSString *myName;  
}
```

```
@property NSString *myName;  
@end
```



Déclaration des propriétés au même endroit que les méthodes

équivalent à

↓ -(NSString *) myName;
-(void) setMyName:(NSString *) newName;



|| Quelques mots sur objective-C...

Propriétés

Implémenter les accesseurs

```
#import "MaClasse.h"
```

```
@implementation MaClasse
```

```
@synthesize myName;
```

```
@end
```

La variable d'instance myName est donc librement accessible :
Modifier sa valeur par la notation . ou par l'envoie de message.

équivalent à

```
- (NSString *) myName  
{  
    return myName;  
}  
  
-(void) setMyName:(NSString *) newName  
{  
    myName=newName;  
}
```

Exemple: [self setMyName:newName]; (mutateur)

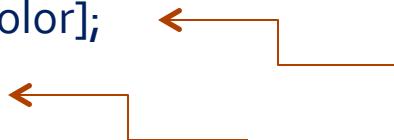


|| Quelques mots sur objective-C...

Exemple 1

Un objet a des accesseurs pour une propriété color

```
UIColor *c = [objet color];  
[objet setColor:c];
```



Méthode qui renvoie la couleur
d'un objet (**accesseur/getter**)

Méthode qui fixe la couleur
d'un objet (**mutateur/setter**)

```
UIColor *c = object.color;  
objet.color=c;
```

Accès à une propriété via la notation point

|| Quelques mots sur objective-C...



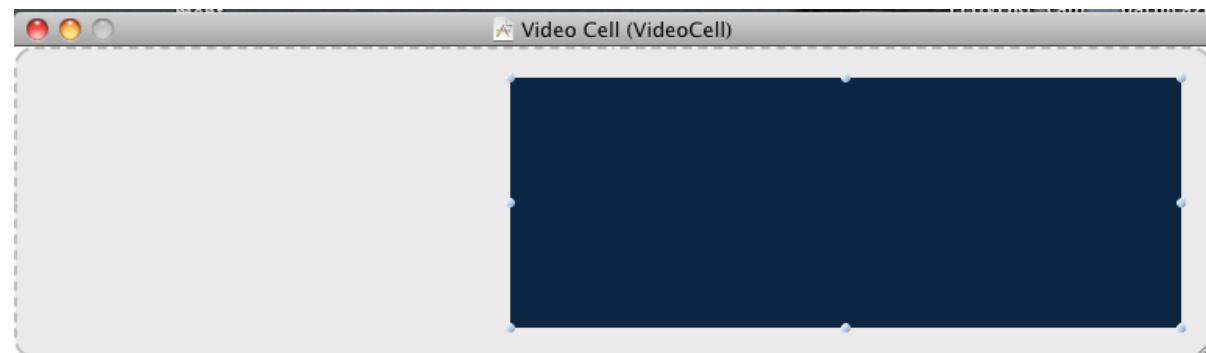
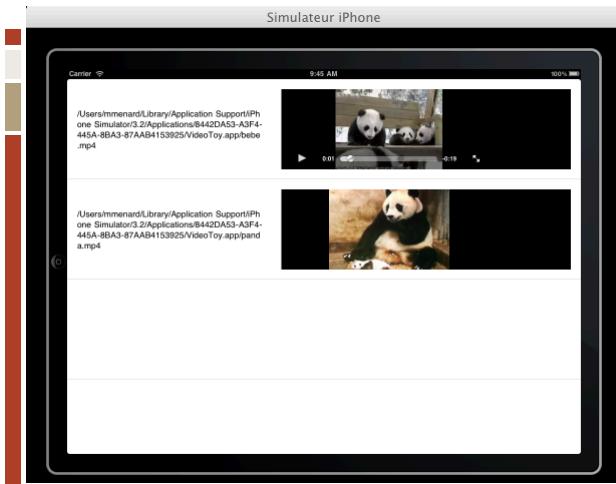
Exemple 2 (Classe VideoCell - .xib)



VideoCell.xib

View Mode Inspector Search Field

Name	Type
File's Owner	VideoToyViewC...
First Responder	UIResponder
Video Cell (VideoCell)	VideoCell
View	UIView
Label	UILabel



||| Quelques mots sur objective-C...



Exemple 2(Classe VideoCell - .h)

```
#import <UIKit/UIKit.h>
#import <MediaPlayer/MediaPlayer.h>

@interface VideoCell : UITableViewCell {
    IBOutlet UIView *movieViewContainer;
    IBOutlet UILabel *urlLabel;
    NSString *urlPath;
    MPMoviePlayerController *mpc;
}

@property (retain, nonatomic) NSString *urlPath;
@property (retain, nonatomic) MPMoviePlayerController *mpc;

+ (NSString *)reuseIdentifier;
+ (CGFloat)rowHeight;

@end
```



|| Quelques mots sur objective-C...

Exemple 2 (Classe VideoCell -.m)

```
#import "VideoCell.h"

@implementation VideoCell
@synthesize urlPath, mpc;

+ (NSString *)reuseIdentifier {
    return @"VideoCell";
}
+ (CGFloat)rowHeight {
    return 200;
}
```



|| Quelques mots sur objective-C...

Exemple 2 (Classe VideoCell -.m)

Création du setter pour la propriété UrlPath

```
- (void)setUrlPath:(NSString *)p {  
    if (![p isEqualToString:urlPath]) {  
        [urlPath autorelease];  
        urlPath = [p retain];  
        if (urlPath && !mpc) {  
            [self setupMpc];  
        }  
        urlLabel.text = urlPath;  
    }  
}
```



Appel de la fonction setupMpc
Lecture à travers la propriété
setUrlPath



||| Quelques mots sur objective-C...

Attributs et propriétés

Il est possible de fournir au compilateur des indications sur la propriété pour changer le code généré

```
// propriété en lecture seule  
@property (readonly) NSString *string;
```

```
// propriété en mode non-atomic  
@property (nonatomic) NSString *string;
```

```
// l'objet assigné à cette propriété doit être retenu  
@property (retain) NSString *string;
```

```
// propriété en mode non-atomic et devant être retenu  
@property (nonatomic,retain) NSString *string;
```

Le code généré pour les propriétés est construit pour être appellable depuis plusieurs threads en même temps.

|| Quelques mots sur objective-C...

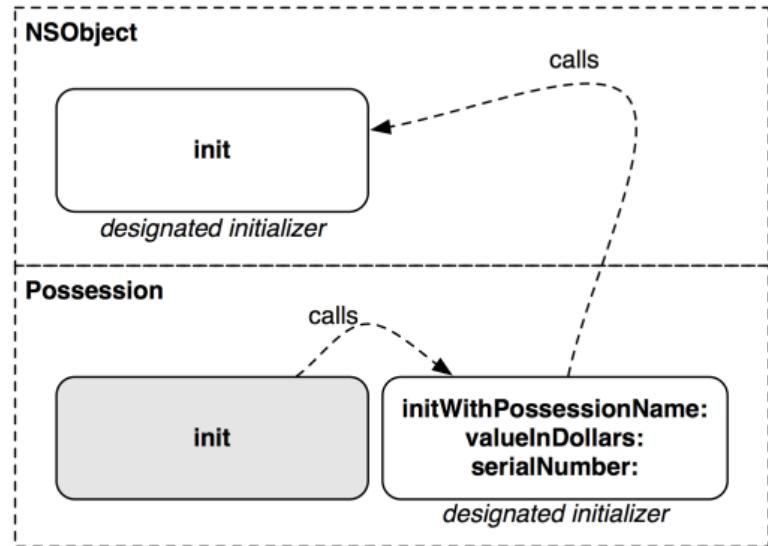


Initialisateur désigné (exemple)

```
- (id)initWithPossessionName:(NSString *)name
                      valueInDollars:(int)value
                        serialNumber:(NSString *)sNumber
{
    // Call the superclass's designated initializer
    self = [super init];

    // Did the superclass's designated initializer succeed?
    if (self) {
        // Give the instance variables initial values
        [self setPossessionName:name];
        [self setSerialNumber:sNumber];
        [self setValueInDollars:value];
        dateCreated = [[NSDate alloc] init];
    }

    // Return the address of the newly initialized object
    return self;
}
```



||| Quelques mots sur objective-C...



La bibliothèque standard : le framework foundation : chaîne

Chaîne de caractère:

Encapsulées dans des objets NSString



Fournit les méthodes nécessaires
-Mesurer la taille
-Concaténer plusieurs chaines
-Retrouver une occurrence

Déclaration

```
NSString *LanguageName=@"Objective-C";
```



Syntaxe @''' : déclaration et initialisation

```
NSString *LanguageName=[NSString stringWithFormat:@"Mon langage est %@ %u »,  
 @"Objective-C",2];
```



||| Quelques mots sur objective-C...

La bibliothèque standard : le framework foundation : chaîne (suite)

La classe `NSString` est non mutable et ses instances ne peuvent être modifiées.
(`NSNumber`, `NSArray`, `NSDictionary`)

Chaîne de caractère non mutable:

On crée une nouvelle instance à partir de l'instance existante.

```
NSString *string=@"Bonjour ";
string=[string stringByAppendingString:@"tout le monde"];
```

On crée une chaîne, puis une deuxième à partir de la première

Chaîne de caractère mutable (attention au mode d'initialisation) :

```
NSMutableString *string=[NSMutableString stringWithString:@"Bonjour "];
[string appendString:@"tout le monde"];
```

Utiliser un objet mutable qu'on initialise et qu'on modifie ensuite



||| Quelques mots sur objective-C...

La bibliothèque standard : le framework foundation : listes

La classe NSArray fournit une abstraction de liste ordonnée de taille fixe

```
NSArray *uneliste=[[NSArray alloc] initWithObjects:obj1,obj2,obj3,nil];
NSObject *obj=[uneList objectAtIndex:2]; //obj 3
```

Initialisation et utilisation d'une liste modifiable

```
NSMutableArray *uneListeModifiable=[[NSMutableArray alloc] initWithCapacity:10];
[uneListeModifiable addObject:obj1];
[uneListeModifiable addObject:obj2];
[uneListeModifiable addObject:obj3];
NSLog(@"%@",@"Taille de la liste: %u",[uneListeModifiable count]);
```



||| Quelques mots sur objective-C...

La bibliothèque standard : le framework foundation : listes (suite)

Parcourir rapidement une liste

```
for (Type *element in array)
{
    // ...
}
```

Exemple : imprimer le contenu et la taille de chaque chaîne dans une liste d'objets NSString

```
for (NSString *uneChaine in chaineListe)
{
    NSLog(@"Chaine: %@ Longueur: %u",uneChaine,[uneChaine length]);
}
```



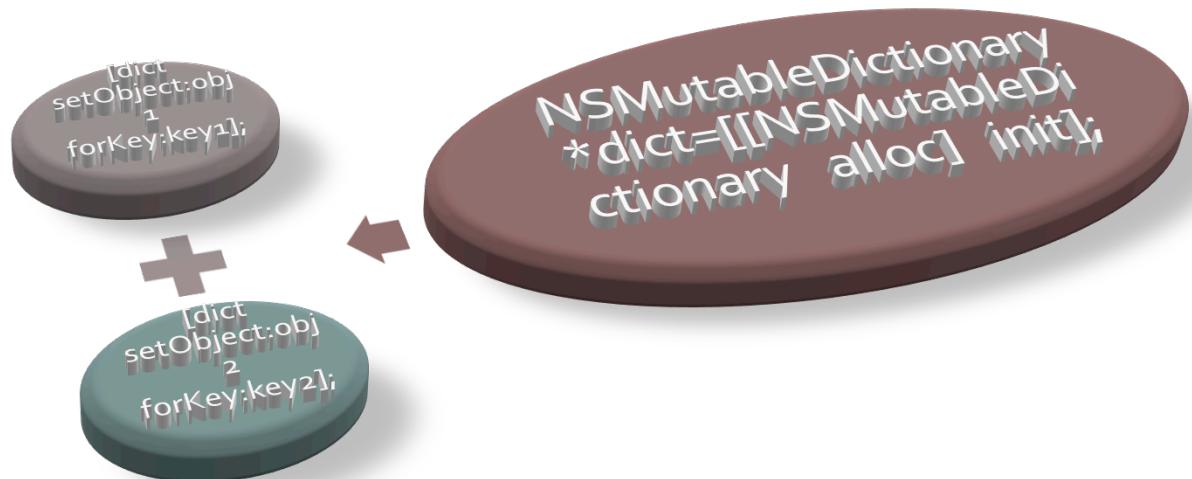
||| Quelques mots sur objective-C...

La bibliothèque standard : le framework foundation : Dictionnaire

La classe NSDictionary est une table de hachage qui pour une clé donnée garde une valeur.

```
NSMutableDictionary *dict=[[NSMutableDictionary alloc] init];
[dict setObject:obj1 forKey:key1];
[dict setObject:obj2 forKey:key2];
```

```
NSObject *o = [dict objectForKey : key2]; // obj2
```



||| Quelques mots sur objective-C...



Le mécanisme de comptage de référence

Pour stocker des objets en mémoire, l'application doit demander au système de lui allouer de la mémoire.

Le système garde une trace de toutes les zones mémoires allouées à l'application, mais ne peut pas savoir lesquelles sont encore utilisées.

L'application doit indiquer au système que les zones mémoires ne sont plus utilisées.

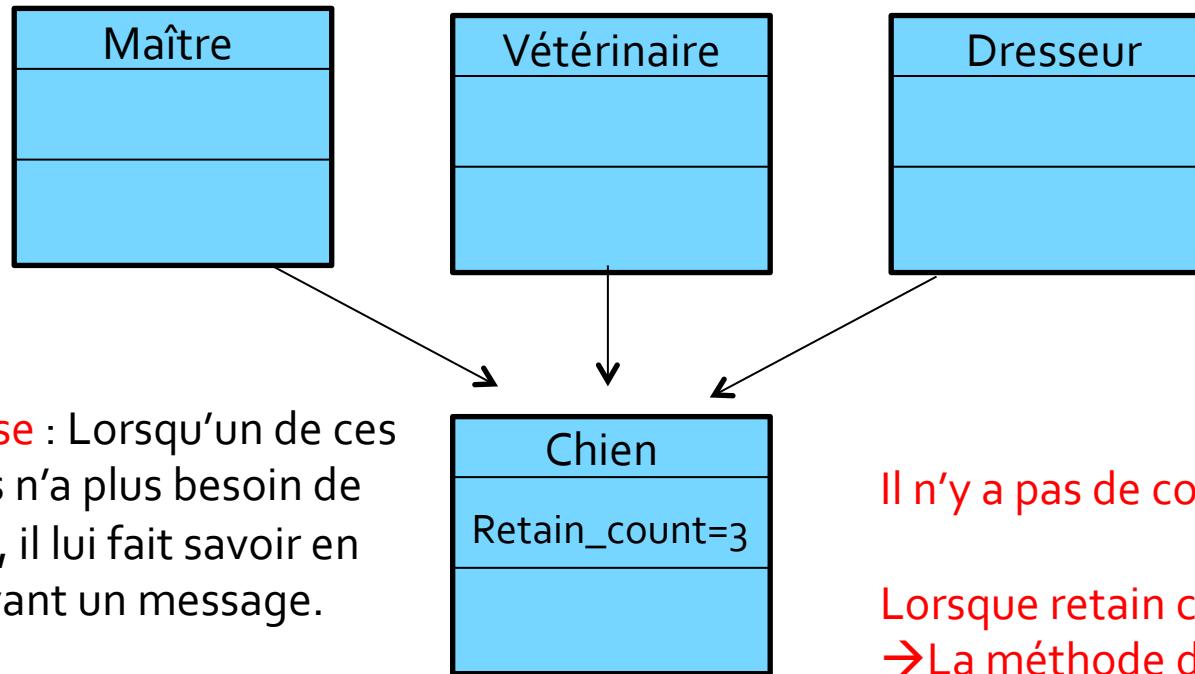


|| Quelques mots sur objective-C...



Le mécanisme de comptage de référence : retain

Chaque objet possède un compteur, le retain count qui va compter le nombre d'objets qui ont besoin de lui.



Release : Lorsqu'un de ces objets n'a plus besoin de Chien, il lui fait savoir en envoyant un message.

Il n'y a pas de copie de l'objet
Lorsque retain count =0
→La méthode d'instance
-dealloc est appelée

|| Quelques mots sur objective-C...



Règles de gestion de la mémoire en Objective-C



Vous devenez propriétaires des objets que vous avez créés en utilisant une méthode dont le nom commence par **alloc**, **new**, ou **contient copy**.

Vous devenez également propriétaire des objets que vous **retenez**.

VOUS DEVEZ LIBERER LA MÉMOIRE DES OBJETS QUE VOUS DETENEZ EN APPELANT LA MÉTHODE RELEASE OU AUTORELEASE.

Dans les autres cas, vous ne devez jamais les appeler.

|| Quelques mots sur objective-C...



Application de la règle

*La classe NSObject définit la méthode alloc qui alloue la mémoire pour un objet.
L'objet doit être initialisé en appelant une méthode init.*

```
NSString *str=[[NSString alloc] init];  
// objet créé par la méthode alloc - vous devez le libérer  
[str release];
```

```
NSString *str=[NSString stringWithFormat:@""];  
// objet créé par une méthode dont le nom ne commence pas par alloc ou new  
// Vous ne devez pas le libérer
```

||| Quelques mots sur objective-C...



Application de la règle et comptage de référence

retain

:

objet retenu

release

:

objet libéré

*allocation :
CR=1 → 1 release*

```
int main (int argc, char *argv[]) {  
    NSLog(@"start");  
    NSString *str=[[NSString alloc] initWithFormat:@"Hello %@ ", @"World"];  
    NSLog(@"Compteurs de références: %u", [str retainCount]);
```

```
[str retain];
```

```
NSLog(@"Compteurs de références: %u", [str retainCount]);
```

```
[str release];
```

```
NSLog(@"Compteurs de références: %u", [str retainCount]);
```

```
[str release];
```

```
NSLog(@"done");
```

start

1

2

1

done

|| Quelques mots sur objective-C...



Envoyer un message à un objet dont la mémoire a été libérée

```
int main (int argc, char *argv[]) {  
    NSString *str=[[NSString alloc] initWithFormat:@"Hello %@ %@", @"World";  
    [str release];  
    [str isEqualToString:@""];  
}
```



EXC_BAD_ACCESS

Le programme a essayé de lire une zone mémoire qui ne lui appartient pas/plus.

Libérer plusieurs fois la mémoire d'un objet → exception !

Revient à appeler une méthode release sur un objet dont la mémoire a déjà été libérée

Release → compteur -1 → appel à dealloc si compteur==0



|| Quelques mots sur objective-C...



Libération retardée d'un objet

```
- (NSString*) giveMeSomething  
{  
    NSString *s=[[NSString] alloc] init];  
    [s autorelease];  
    return s;  
}
```

Méthode

- Renvoie un objet
- l'appelant n'a pas à se soucier de le libérer

Autorelease : permet de dire à l'objet de se libérer un peu plus tard.

La classe appelante peut récupérer l'objet et l'utiliser sans se soucier de le libérer.

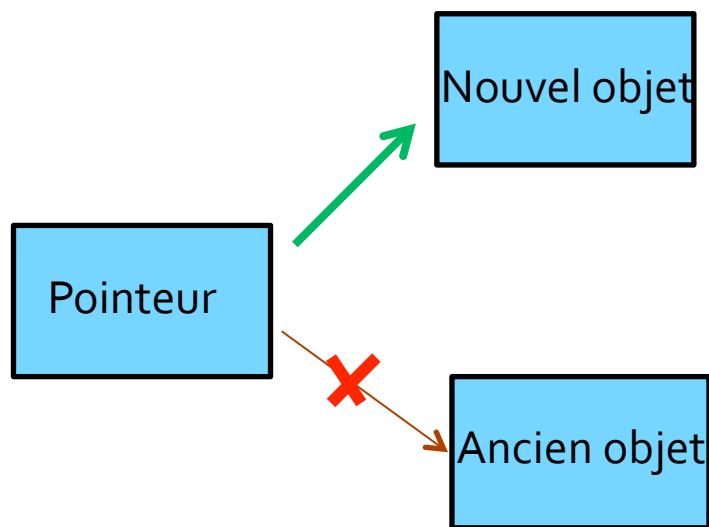
Par contre, il ne peut pas être stocké dans une variable d'instance et réutilisé plus tard puisque libéré par le pool d'autorelease : il faut donc le retenir.

```
str=[NSString stringWithFormat:@""];  
[str retain];
```

|| Quelques mots sur objective-C...



Copier et **assigner**



Le principe de l'assignment

- C'est changer l'objet où va pointer ce pointeur
- Utiliser dans les accesseurs

|| Quelques mots sur objective-C...



Assigner et accesseurs

Exemples de codes

Classe Personnage : possède une variable arme
de type Epee : on souhaite changer la valeur de arme

```
-(void) setArme : (Epee *) nouvelleArme
{
    arme=nouvelleArme;
}
```

Code non correcte !

Le principe de l'assignation

- C'est changer l'objet où va pointer ce pointeur
- Utiliser dans les accesseurs

Deux pointeurs se trouvent sur le même objet.

L'objet initial se lequel pointait arme se retrouve orphelin → fuite mémoire.

|| Quelques mots sur objective-C...



Assigner et accesseurs

Exemples de codes

Classe Personnage : possède une variable arme
de type Epee : on souhaite changer la valeur de arme

```
- (void) setArme : (Epee *) nouvelleArme
{
    [arme release];
    arme=nouvelleArme;
}
```

Code non correcte !

Le principe de l'assignation

- C'est changer l'objet où va pointer ce pointeur
- Utiliser dans les accesseurs

Exemple:

L'objet pointé par nouvelleArme est partagé par un autre pointeur
→ retain count =2

Exécution de setArme → retain count=2 !
(et pourtant 3 objets utilisent nouvelleArme →
L'objet sera libéré sans l'accord de Personnage)

|| Quelques mots sur objective-C...



Assigner et accesseurs

Exemples de codes

Classe Personnage : possède une variable arme
de type Epee : on souhaite changer la valeur de arme

```
-(void) setArme : (Epee *) nouvelleArme
{
    [arme release];
    [nouvelleArme retain];
    arme=nouvelleArme;
}
```

Code non correcte !

Le principe de l'assignation

- C'est changer l'objet où va pointer ce pointeur
- Utiliser dans les accesseurs

Exemple:

Si les deux pointeurs arme et nouvelleArme pointent sur le même objet avant l'exécution de setArme → si on envoie un release à arme, c'est comme si il était envoyé à nouvelleArme.

Il faut donc vérifier que les deux pointeurs ne pointent pas sur le même objet.

|| Quelques mots sur objective-C...



Assigner et accesseurs

Exemples de codes

Classe Personnage : possède une variable arme
de type Epee : on souhaite changer la valeur de arme

```
-(void) setArme : (Epee *) nouvelleArme
{
    if (arme!=nouvelleArme) {
        [arme release];
        [nouvelleArme retain];
        arme=nouvelleArme;
    }
}
```

Code correcte !

Le principe de l'assignation

- C'est changer l'objet où va pointer ce pointeur
- Utiliser dans les accesseurs

Remarque:

On a un pointeur qui pointe vers cet objet.
Mais s'il y a d'autres pointeurs sur cet objet et qu'il est modifié de l'extérieur, il sera aussi modifié dans l'objet.

Si on ne souhaite pas ce comportement → utiliser la méthode d'instance - copy

|| Quelques mots sur objective-C...



Copier et assigner

Classe Personnage : possède une variable arme de type Epee : on souhaite changer la valeur de arme

```
-(void) setArme : (Epee *) nouvelleArme
{
    if (arme!=nouvelleArme) {
        [arme release];
        arme=[nouvelleArme copy];
    }
}
```

Code correcte !

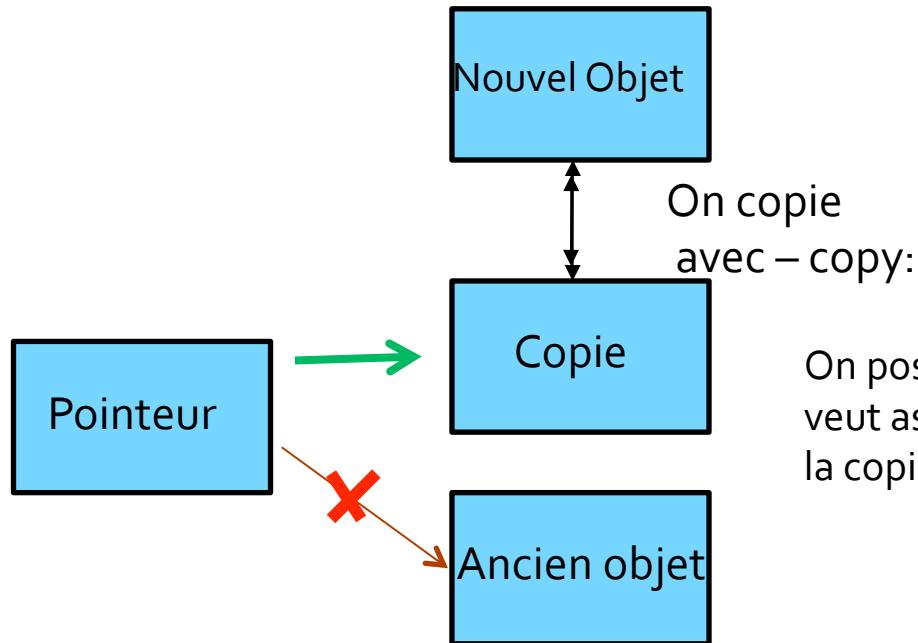
Le principe de la copie

- L'objet doit être conforme au protocole NSCopying.
- - copy augmente le retain count de 1

|| Quelques mots sur objective-C...



Copier et assigner



On possède une copie de l'objet que l'on veut assigner. Si « Nouvel Objet » est modifié, la copie ne le sera pas.

Le principe de la copie

- L'objet doit être conforme au protocole `NSCopying`.
- `-copy` augmente le retain count de 1

|| Quelques mots sur objective-C...



Pour une propriété, utilisez les accesseurs ou la notation point.

```
NSString *aString=[[NSString alloc] init];  
self.maString=aString;  
[aString release];
```

Méthode

- Allocation d'un objet
- Assigner l'objet à une propriété
- Libérer l'objet

-Il est d'usage de l'assigner à une variable locale, puis de l'affecter à la propriété
→ *ce qui entraîne une incrémentation du compteur de références*
avant de le libérer

|| Quelques mots sur objective-C...

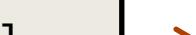


Libérer la mémoire

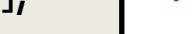
```
-(void) dealloc  
{  
    [epee release];  
    [bouclier release];  
    [super dealloc];  
}
```



Libération manuelle



Libération manuelle

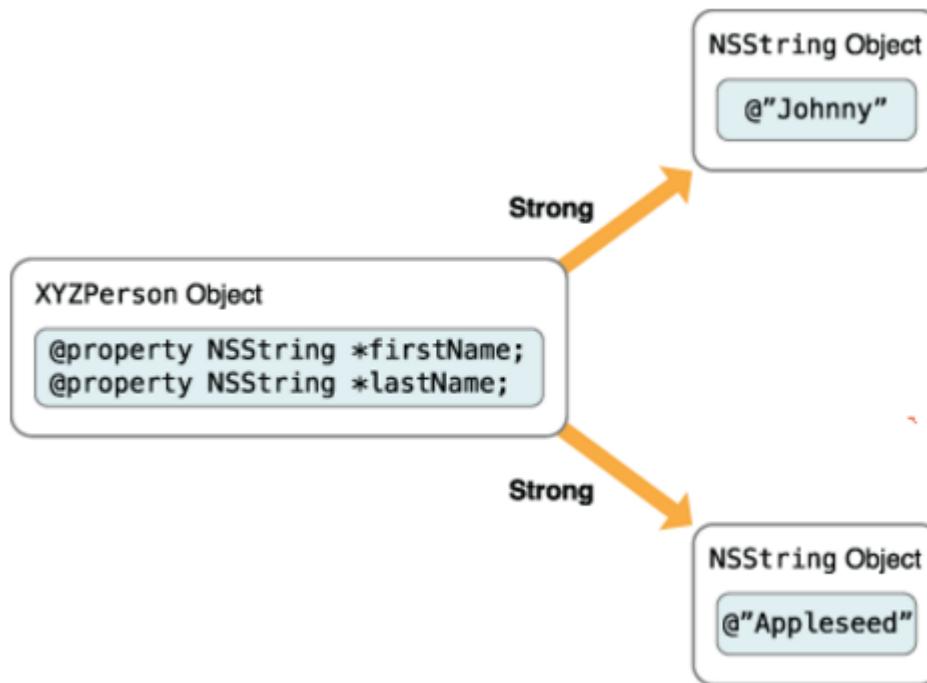


Toutes les variables membres de Type int, char, float, etc seront libérées par cette méthode.
mot-clé super symbolise une instance de la classe mère : on peut donc exécuter la méthode -dealloc de la classe mère.

|| Quelques mots sur objective-C...



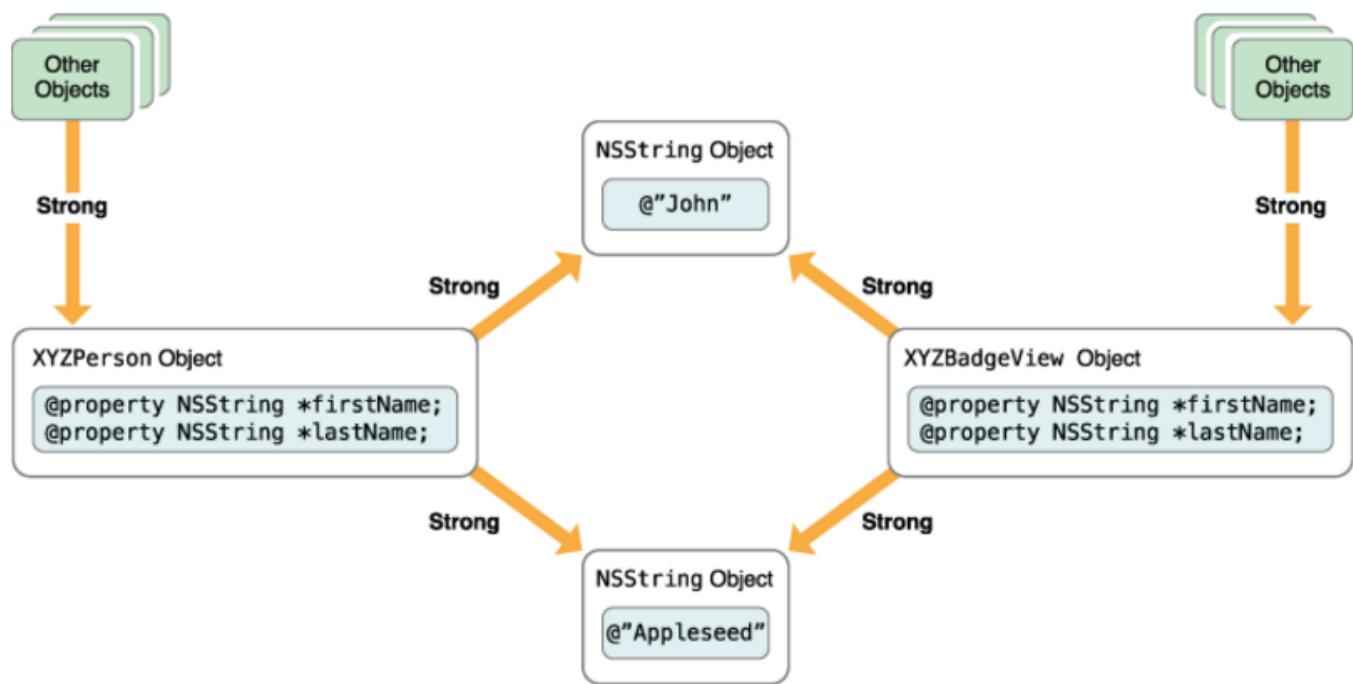
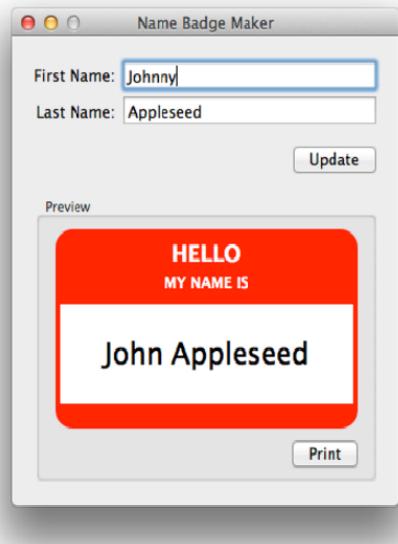
Libération gérée automatiquement (relation strong)



|| Quelques mots sur objective-C...



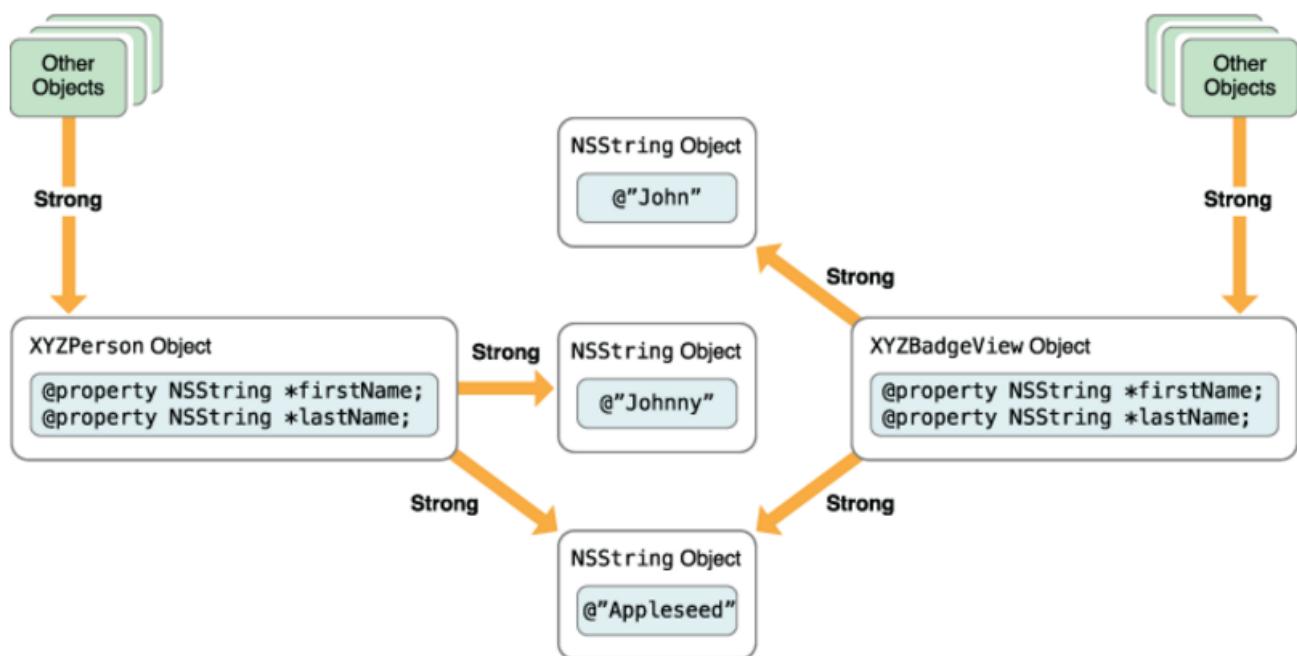
Libération gérée automatiquement (relation strong)



|| Quelques mots sur objective-C...



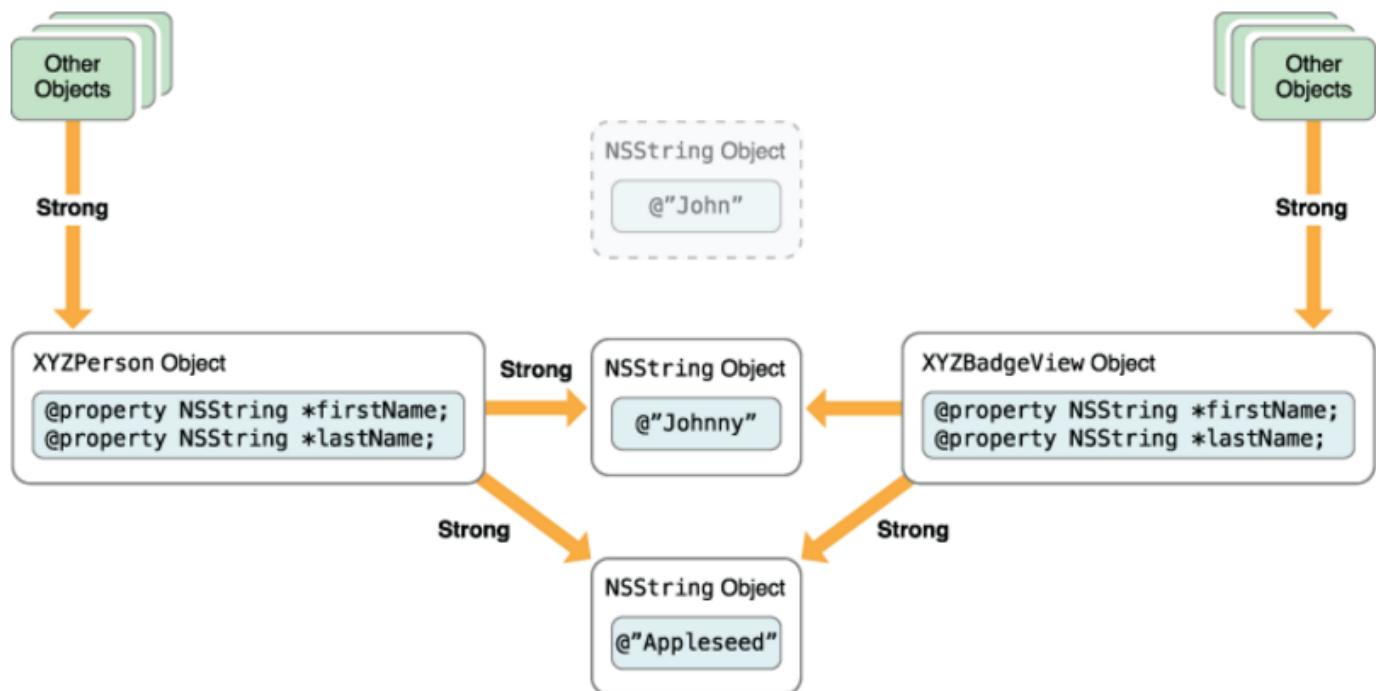
Libération gérée automatiquement (relation strong)



|| Quelques mots sur objective-C...



Libération gérée automatiquement (relation strong)





Complément sur objective C

Protocole et **gestion mémoire**



Illustrations / Applications

Constructeur de commodité

Bassin d'autorelease

Getter

Initialisation à partir d'un objet existant

Mutateur

Tableau et gestion mémoire



Complément sur objective C

Protocole et **gestion mémoire**



Libérer la mémoire : propriétaire d'un objet ou pas !

Vous devez envoyer release à cet objet

```
NSString * chaine_1 = [[NSString alloc] initWithString:@"Vous êtes le propriétaire "];  
NSString * chaine_2 = [NSString stringWithFormat:@"Vous n'êtes pas le propriétaire "];
```

Constructeur de commodité : même rôle que init mais il fait le alloc lui-même.

L'objet envoyé est inscrit au bassin d'autorelease, et sera donc temporaire, s'il ne lui est pas envoyé un retain.

Le nom devrait toujours être préfixé par celui de la classe



Complément sur objective C

Protocole et **gestion mémoire**



Ecriture d'un constructeur de commodité : constructeur virtuel

```
@implementation Vehicule  
+(id) vehiculeDeCouleur:(NSColor*)couleur  
{  
  
    id nouvelleInstance = [[[self class] alloc] init];  
  
    [nouvelleInstance setCouleur:couleur];  
    return [nouvelleInstance autorelease];  
}  
@end  
@interface Voiture : Vehicule {...}  
@end  
...  
//produit une voiture (rouge) !  
id voiture = [Voiture vehiculeDeCouleur:  
[NSColor redColor]];
```

Utilisation du faux mot-clé `class` renvoyant l'objet classe de l'objet courant : instance de la métaclass (méthode de `NSObject`)

La classe est identifiée dynamiquement

Complément sur objective C

Protocole et gestion mémoire



Les bassins d'autorelease

```
#import <Foundation/Foundation.h>
Int main (int argc, const char * argv[])
{
    NSAutoreleasePool *pool = [[NSAutoreleasePool alloc] init];
    NSLog("Hello, World!");
    [pool drain];
    return 0;
}
```

Le rôle de l'autorelease est d'envoyer le message release à chacun des objets dans le bassin à un certain moment.

Pour ajouter un objet à un objet NSAutorelease, il faut lui envoyer le message autorelease.

Complément sur objective C

Protocole et gestion mémoire



Les bassins d'autorelease : utilisation dans un accesseur de type getter

```
- (Bouclier *) bouclier  
{  
    return bouclier;  
}
```

L'objet retourné n'appartient pas à celui qui appelle le Getter → il risque d'être libéré avant que l'utilisateur de l'objet ait terminé (référencement faible)

```
- Objet = [[guerrier bouclier] retain];
```

Complément sur objective C

Protocole et gestion mémoire



Les bassins d'autorelease : utilisation dans un accesseur de type getter

```
-(Bouclier *) bouclier
{
    [bouclier retain];
    [bouclier release];
    return bouclier;
}
```

Code non correcte !

On prend en charge l'objet
On se décharge de l'objet
Cela ne sert à rien !!

```
-(Bouclier *) bouclier
{
    [bouclier retain];
    return bouclier;
    [bouclier release];
}
```

Code non correcte !

On prend en charge l'objet
On le renvoie
Cette ligne ne sera jamais exécutée !!

Complément sur objective C

Protocole et gestion mémoire



Les bassins d'autorelease : utilisation dans un accesseur de type getter

```
-(Bouclier *) bouclier
{
    [bouclier retain];
    [bouclier autorelease];
    return bouclier;
}
```

Code correcte !

```
-(Bouclier *) bouclier
{
    return [[bouclier retain] autorelease];
}
```

Code correcte !

Relativement lourde en mémoire : si votre getter est appelé très fréquemment, utilisez la première implémentation.

Complément sur objective C

Protocole et gestion mémoire



Créer un objet à partir d'un autre

Méthode d'initialisation

```
- (void) initWithColor: (NSColor *) couleur
{
    self=[super init];
    if(self !=nil)
        maCouleur=[couleur retain];
        return self;
    }
    else
        return nil;
}
```

Complément sur objective C

Protocole et gestion mémoire



Créer un tableau

```
NSArray * tableau = [[ NSArray alloc] init];  
/* Et quand on a fini ... */  
[tableau release];
```

En utilisant une méthode de classe

```
NSArray * tableau = [[ NSArray array];
```

Complément sur objective C

Protocole et gestion mémoire



Créer un tableau : on utilise ces méthodes pour initialiser à partir d'autres objets ou en utilisant les objets que doit contenir le NSArray que l'on veut créer.

```
/* A partir d'un autre NSArray */  
NSArray * tableau = [[ NSArray alloc] initWithArray: unAutreTableau];  
NSArray * tableau = [NSArray arrayWithArray: unAutreTableau];
```

```
/* Avec un seul objet */  
NSArray * tableau = [[ NSArray alloc] initWithObject: unObjet];  
NSArray * tableau = [NSArray arrayWithObject: unObjet];
```

```
/* Avec plusieurs objets */  
NSArray * listeDesCourses= [NSArray arrayWithObjects: @"dentifrice", @"papier",nil];  
NSArray * tachesaFaire = [NSArray arrayWithObject:listeDesCourses,@" laver voiture",nil];
```

Complément sur objective C

Protocole et gestion mémoire



Créer un tableau : on peut mettre un tableau dans un autre, même si les objets ne sont pas du même type.

```
NSArray * tableau= [NSArray arrayWithObjects:[NSNumber numberWithInt:5],  
[NSNumber numberWithInt:3], nil];
```

```
NSArray * tableau= [NSArray arrayWithObjects:objet1, [NSNull null], objet3, nil];
```

Complément sur objective C

Protocole et gestion mémoire



Parcourir un tableau.

```
NSArray * tableau= [NSArray arrayWithObjects:@"savon",@"Café",@"Chocolat ",nil];  
int nbObjets=[tableau count];
```

```
int i;  
for (i=0;i<[tableau count] ; i++)  
    id objet=[tableau objectAtIndex:i];  
    [objet faireQuelqueChose];  
}
```

-indexForObject: elle retourne l'index d'un objet donné.

Complément sur objective C

Protocole et gestion mémoire



Comparaison dans un tableau.

```
/* Pour obtenir l'index d'un objet */  
int positionHierarchie =[listeDesEmployees indexOfObjects: toto];  
  
/* pour vérifier qu'un objet est dans NSArray */  
If ([listeDesEmployees indexOfObject:tata] != NSNotFound) {  
// l'objet existe  
}  
}
```

-indexOfObject: La fonction renvoie un int qui vaut NSNotFound si l'objet ne se trouve pas dans le NSArray.

Complément sur objective C

Protocole et gestion mémoire

Modifier un tableau.



```
NSMutableArray * tableau = [NSMutableArray array];
[tableau addObject:@"Do"];
[tableau insertObject:@"Ré" atIndex:0];
/* On a : @"Ré", @"Do" */

[tableau replaceObjectAtIndex:1 withObject:@"Mi"];
[tableau addObject:@"Fa"];

[tableau removeObjectAtIndex:0];
/* On a : @"Mi", @"Fa" */

[tableau removeAllObjects]; // On vide tout le tableau
```

- Hérite de NSArray
- -addObject:
Ajoute à la fin du tableau
- -insertObject:atIndex:
Ajoute "au milieu" du tableau
- -replaceObjectAtIndex:
withObject:
- -removeObjectAtIndex:
supprime du tableau
- -removeLastObject:
supprime à la fin du tableau

Complément sur objective C

Protocole et gestion mémoire



La mémoire avec NSArray

Chaque objet du tableau se retrouve retenu par celui-ci

→ Vous pouvez donc envoyer le message –release à un objet ajouté dans un NSArray sans le libérer de la mémoire.

Une fois que NSArray est libéré, il envoie le message –release à tous ses objets.

```
id monObjet = [tableau objectAtIndex:3];  
[tableau release];
```

→ Création d'un pointeur vers un des objets du tableau.
Après libération du tableau, le pointeur est invalide !

Complément sur objective C

Protocole et gestion mémoire



La mémoire avec NSArray

Pour être sûr de ne pas perdre l'objet, on s'occupe nous-mêmes de la mémoire

```
id monObjet = [[tableau objectAtIndex:3] retain];  
  
[tableau release]; *  
  
/* Et quand on a fini */  
[monObjet release];
```

Remarque : La même chose pour NSMutableArray, mais aussi pour les méthodes qui suppriment un objet particulier, comme – removeObjectAtIndex:, car un objet supprimé par ce type de méthode se voit aussi envoyé un message -release

Complément sur objective C

Protocole et gestion mémoire



```
// Vous n'êtes pas propriétaire de enumerator  
NSEnumerator * enumerator = [tableau objectEnumerator];  
id objet;  
  
while (objet = [enumerator nextObject]) {  
    // On boucle tant que la méthode ne renvoie pas nil (ce qui casse la  
    condition)  
  
    /* On travaille avec objet */  
    [objet foo];  
}
```

Chaque objet du type `NSEnumerator` est unique au tableau qu'il parcourt. Les objets `NSArray`, `NSString`, `NSSet` possèdent une méthode appelée `-objectEnumerator` qui retourne un objet `NSEnumerator` qui permet de parcourir le tableau grâce à sa méthode `-nextObject`.

(Retient les objets en mémoire lors de l'énumération)

*Ce que vous devez
(s)avoir avant de
commencer ...*

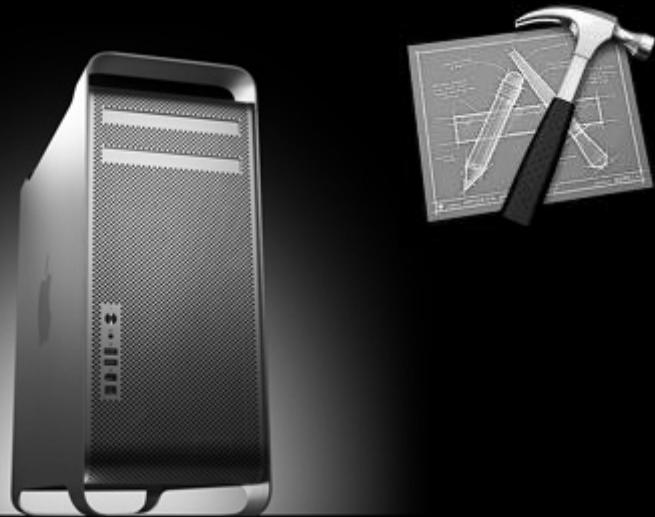
*Quelques mots
sur objective-C
Patterns
essentiels*



*Premiers projets:
Hello
Application
interactive*

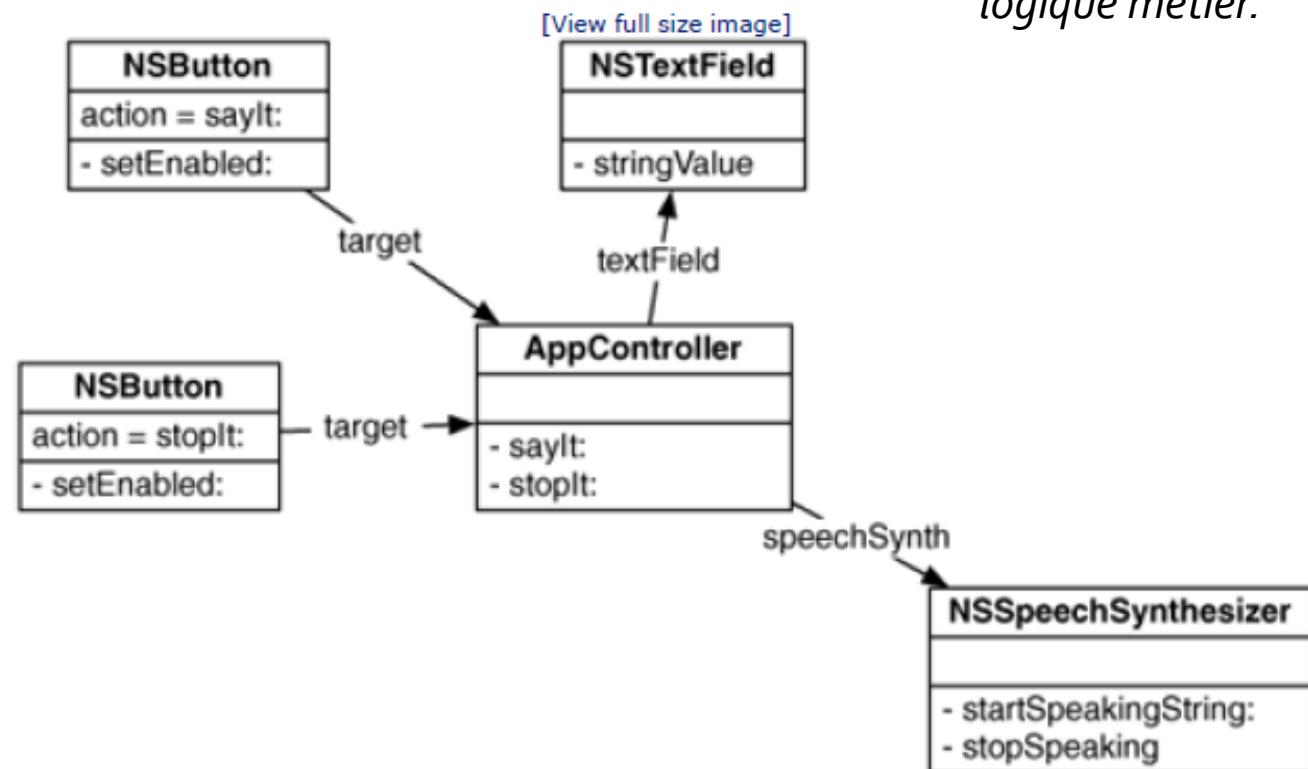
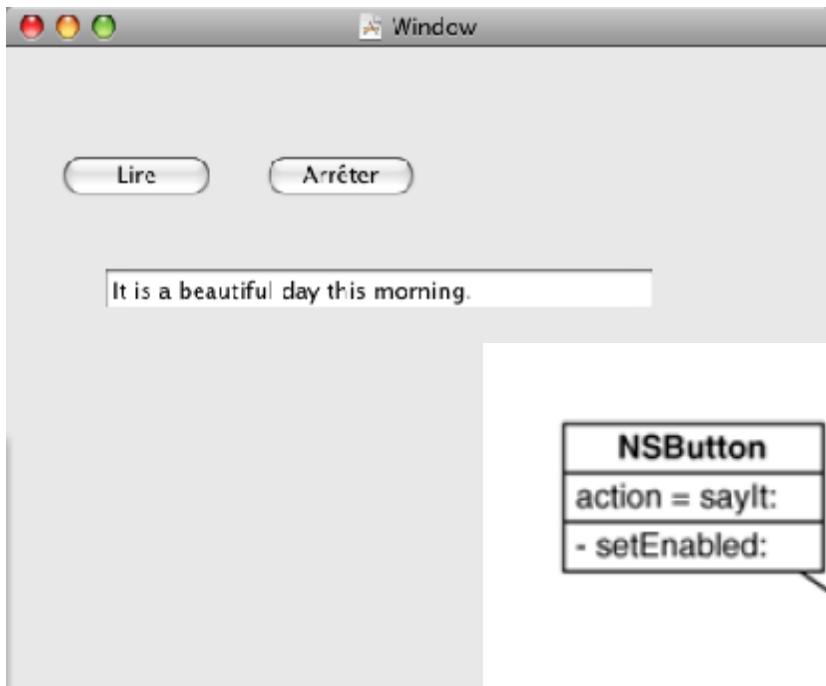
*Plus loin...
Autorotation
Photo
Audio, vidéo*

- Préambule : un exemple de programme sous COCOA



Synthétiseur vocal

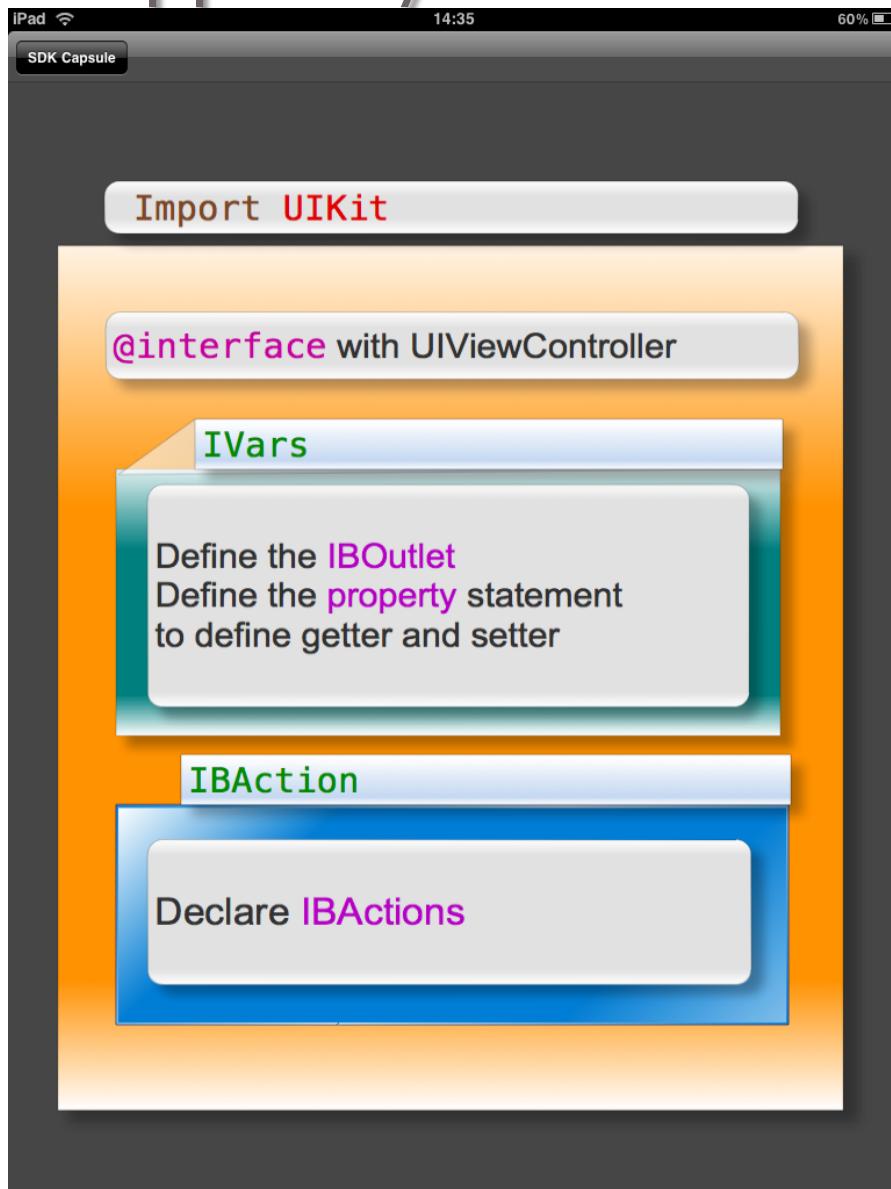
||| Préambule...



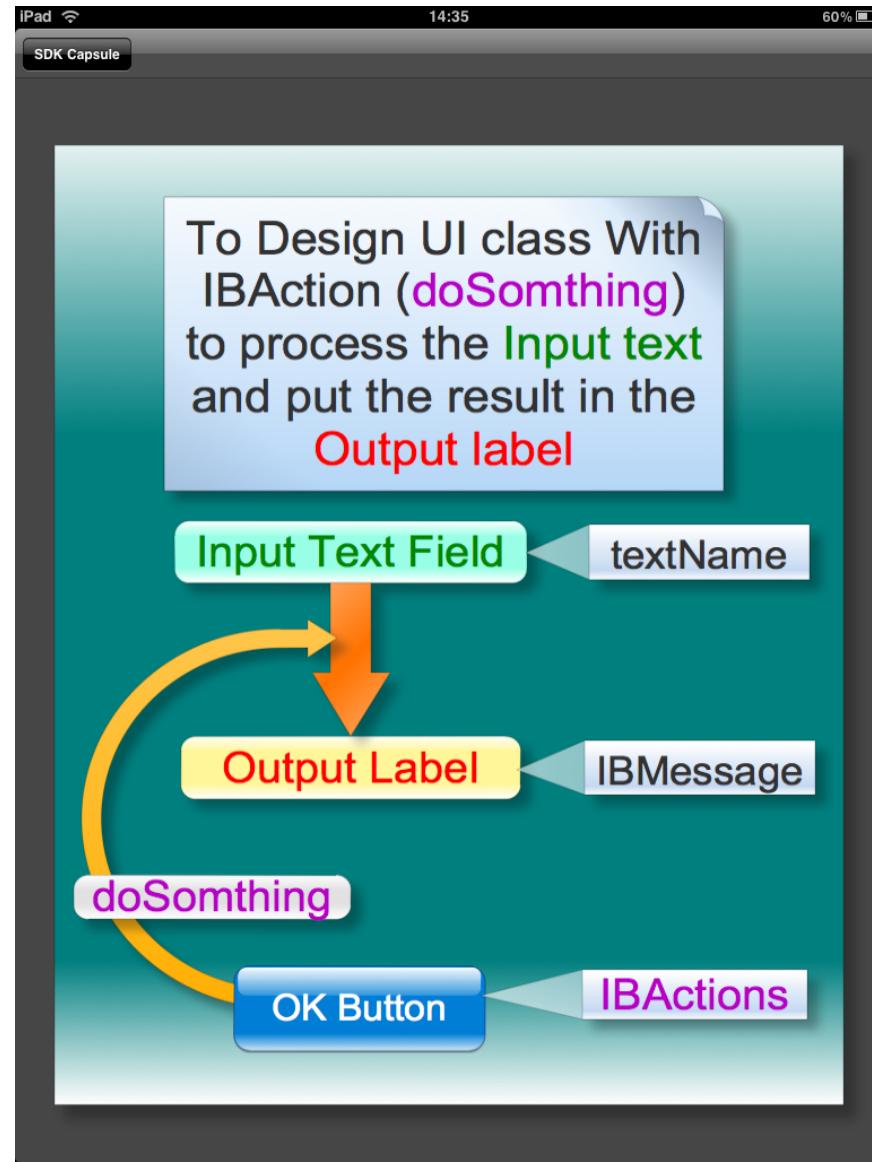
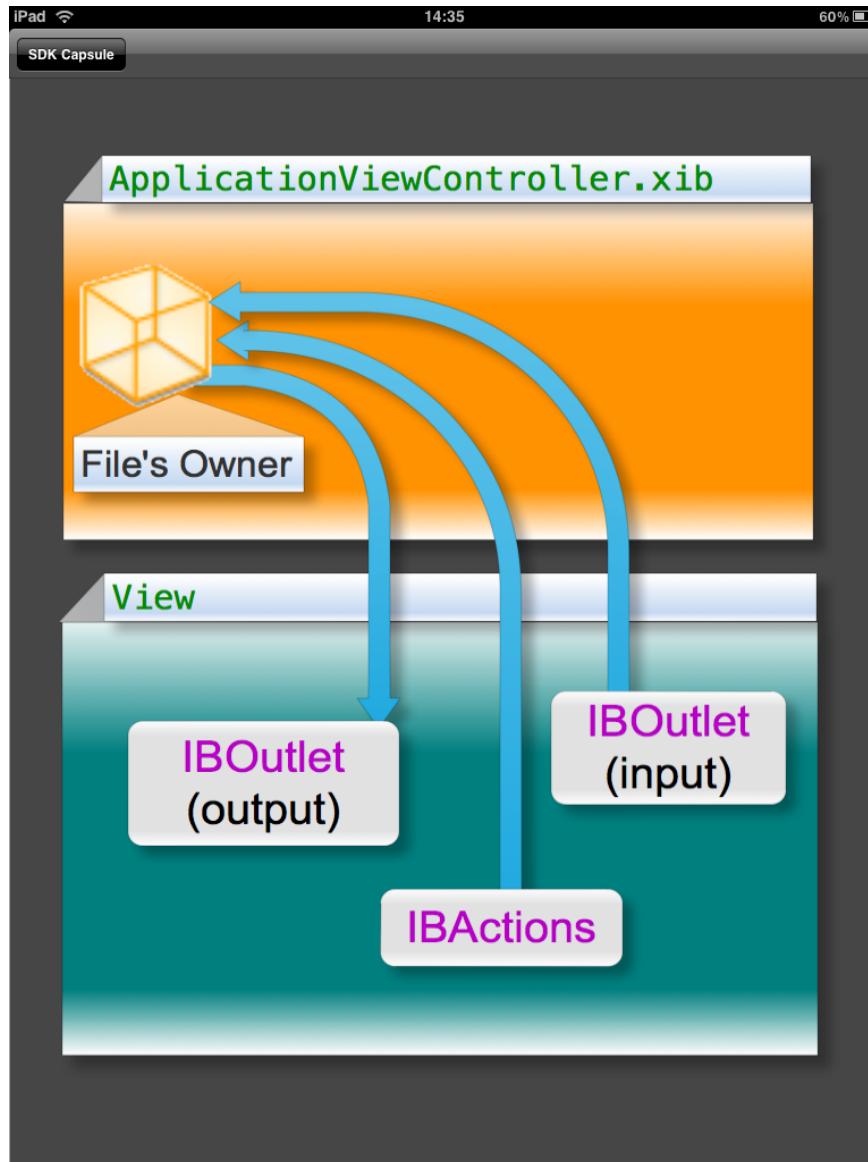
1 contrôleur pour chaque vue

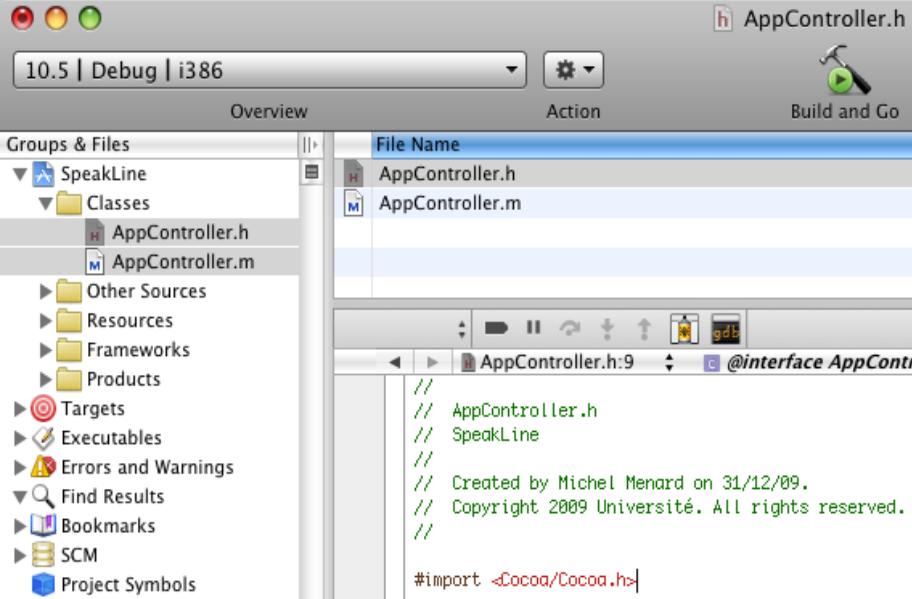
- crée la vue
- charge le modèle
- ajoute un peu de logique métier.

Rappel - synthèse



Rappel - synthèse





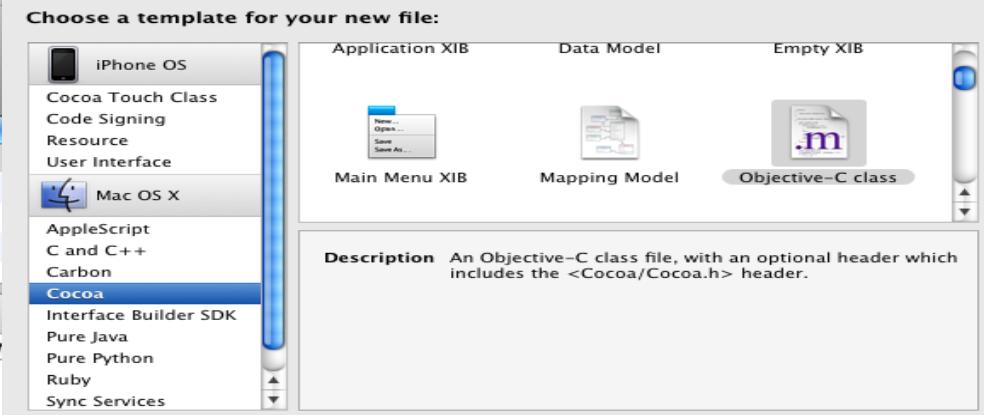
```
#import <Cocoa/Cocoa.h>
```

Création d'un pointeur textField en lien avec l'objet de la vue (IBOutlet)

```
@interface AppController : NSObject
{
    IBOutlet NSTextField *textField;
    NSSpeechSynthesizer *speechSynth;
}
-(IBAction) sayIt:(id) sender;
-(IBAction) stopIt:(id) sender;
```

Méthodes de type action

Interface : décrit le comportement de la classe tel qu'il sera vu par les autres classes.



Implémentation: logique interne de la classe

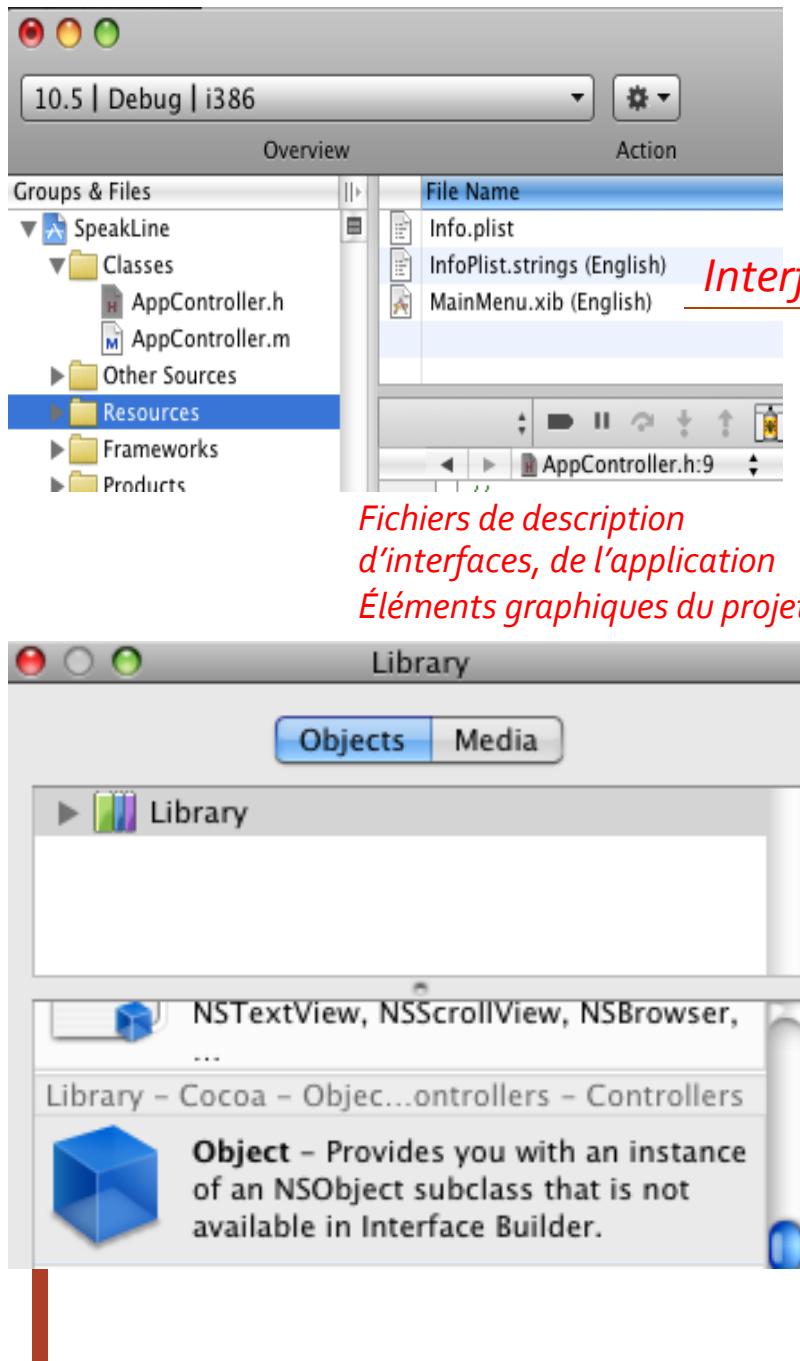
```
#import "AppController.h"

@implementation AppController

- (id) init
{
    [super init];
    NSLog(@"init");
    speechSynth=[[NSSpeechSynthesizer alloc] initWithVoice:nil]; ←
    return self;
}
- (IBAction) sayIt:(id)sender
{
    NSString *string=[textField stringValue]; ←
    if([string length]==0) {
        NSLog(@"La chaîne de %@ est vide",textField);
        return;
    }
    [speechSynth startSpeakingString:string];
    NSLog(@"Lecture de : %@",string);
}

Corps des deux méthodes Action
- (IBAction) stopIt:(id)sender
{
    NSLog(@"Arrêt en cours");
    [speechSynth stopSpeaking]; ←
}

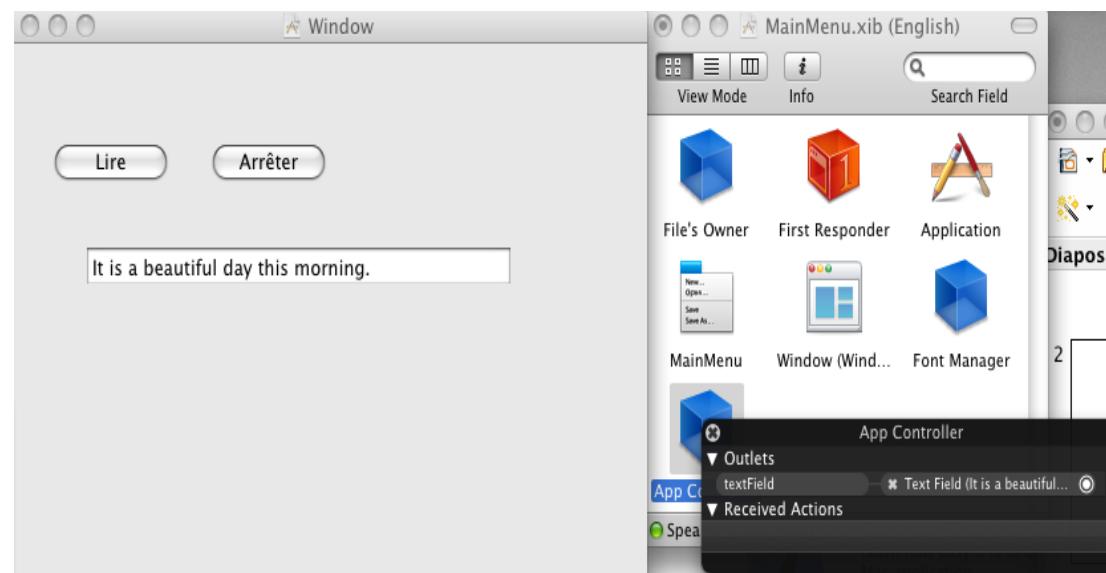
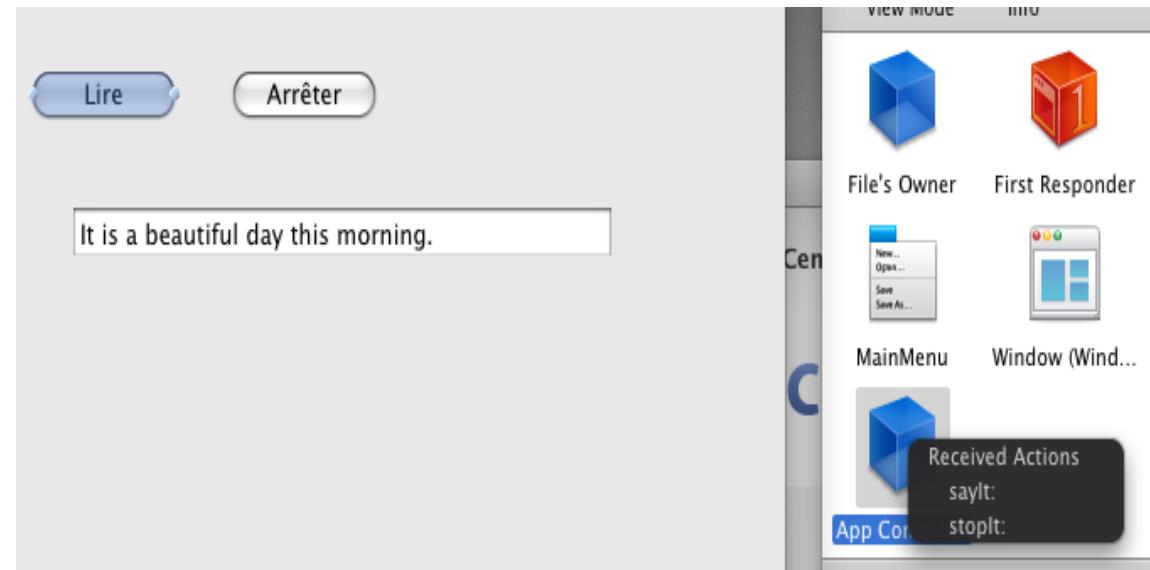
@end
```



Text Field – Displays text that the user can select or edit and that sends its action message to its target when...

Push Button – Intercepts mouse-down events and sends an action message to a target object when it's clicked or...





NSSpeechSynthesizerDelegate

The `NSSpeechSynthesizerDelegate` protocol defines the optional methods implemented by delegates of `NSSpeechSynthesizer` objects.

Language

Swift | Objective-C

SDK

macOS 10.6+

On This Page

Symbols 

Relationships 

Symbols

Synthesizing Speech

`speechSynthesizer:willSpeakWord:ofString:`

Sent just before a synthesized word is spoken through the sound output device.

`speechSynthesizer:willSpeakPhoneme:`

Sent just before a synthesized phoneme is spoken through the sound output device.

`speechSynthesizer:didEncounterErrorAtIndex:ofString:message:`

Sent to the delegate when a speech synthesizer encounters an error in text being synthesized.

`speechSynthesizer:didEncounterSyncMessage:`

Sent to the delegate when a speech synthesizer encounters a synchronization error.

`speechSynthesizer:didFinishSpeaking:`

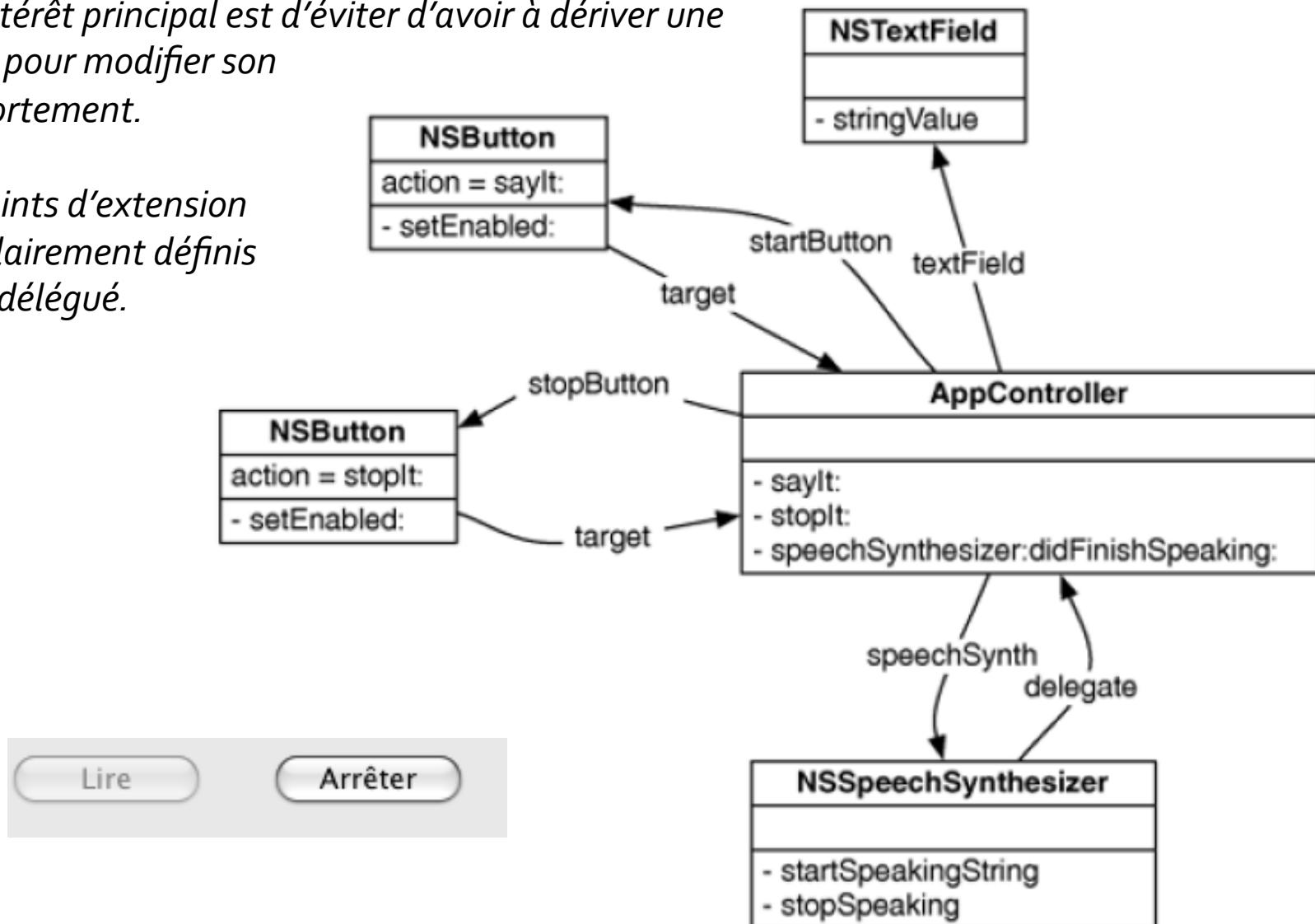
Sent when an `NSSpeechSynthesizer` object finishes speaking through the sound output device.

Délégation de contrôle : permet à une classe de s'appuyer

sur une autre pour enrichir son propre comportement.

Son intérêt principal est d'éviter d'avoir à dériver une classe pour modifier son comportement.

Les points d'extension sont clairement définis par le délégué.



III Complément sur objective C

Protocole et gestion mémoire



Protocoles

L'héritage multiple étant impossible en Objective-C, un mécanisme est ajouté permettant de définir des protocoles qu'une classe s'engage à respecter.

```
@protocol MonProtocole  
- (void)uneMethodeDuProtocole  
  
@optional  
// Déclaration des méthodes optionnelles  
@end
```

Définition d'un protocole dans un fichier d'entête : ressemble à la déclaration d'une classe mais sans variable d'instance.

```
@interface MatroisiemeClasse: NSObject <MonProtocole, MonAutreProtocole>{  
// Variables d'instance  
}  
  
@end
```

Une classe déclare dans son interface la liste des protocoles qu'elle implémente.

```

#import "AppController.h"

@implementation AppController

- (id) init
{
    [super init];
    NSLog(@"init");
    speechSynth=[[NSSpeechSynthesizer alloc] initWithVoice:nil];
    [speechSynth setDelegate:self];
    return self;
}
- (void) speechSynthesizer:(NSSpeechSynthesizer *) sender didFinishSpeaking:(BOOL)complete
{
    NSLog(@"Fin =%d",complete);
    [stopButtonsetEnabled:NO];
    [startButtonsetEnabled:YES];
}
-(IBAction) sayIt:(id)sender
{
    NSString *string=[textField stringValue];
    if([string length]==0) {
        NSLog(@"La chaîne de %@ est vide",textField);
        return;
    }
    [speechSynth startSpeakingString:string];
    NSLog(@"Lecture de : %@",string);
    [stopButtonsetEnabled:YES];
    [startButtonsetEnabled:NO];
}

-(IBAction) stopIt:(id)sender
{
    NSLog(@"Arrêt en cours");
    [speechSynth stopSpeaking];
}

@end

```

Définition du délégué

Cette méthode est appelée lorsque la lecture du texte est terminée (classe NSSpeechSynthesizer)

NSSpeechSynthesizer envoie le message didFinishSpeaking:(BOOL) complete

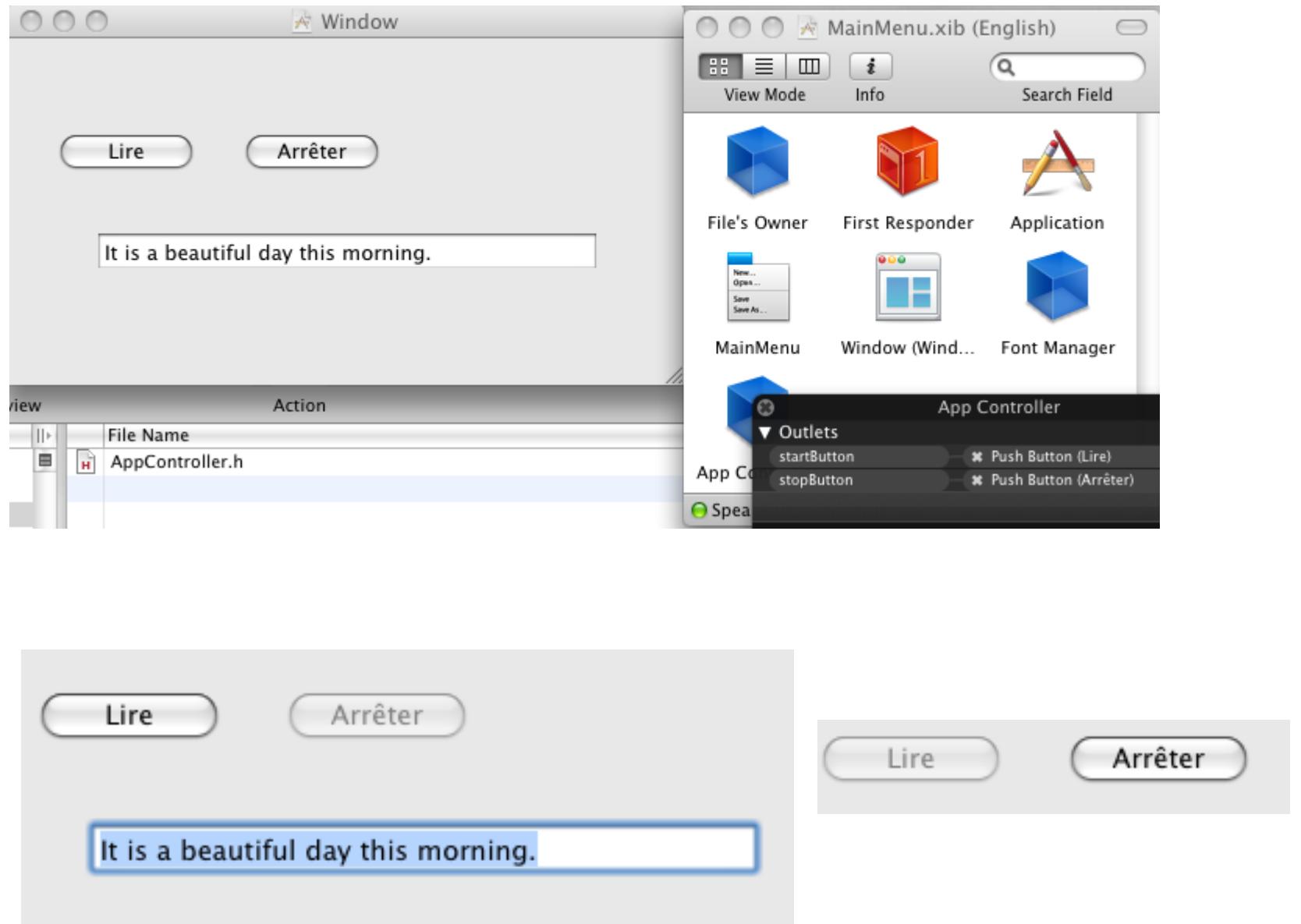
```
#import <Cocoa/Cocoa.h>
```

```

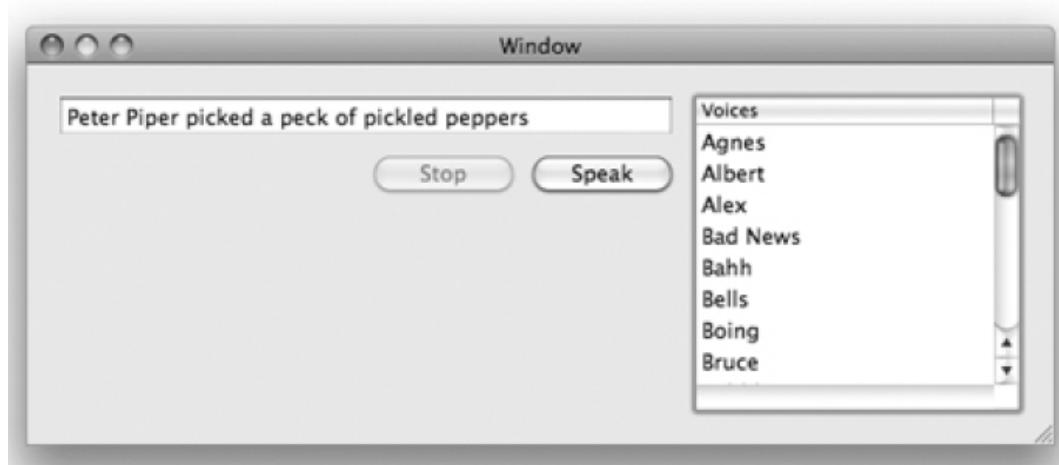
@interface AppController : NSObject {
    IBOutlet NSTextField *textField;
    IBOutlet NSButton *stopButton;
    IBOutlet NSButton *startButton;
    NSSpeechSynthesizer *speechSynth;
}
-(IBAction) sayIt:(id) sender;
-(IBAction) stopIt:(id) sender;

@end

```



DataSource



NSTableViewDataSource

The `NSTableViewDataSource` protocol declares the methods that an instance of `NSTableView` uses to provide the data to a table view and allow editing of the contents of its data source object.

Language

Swift | Objective-C

SDK

macOS 10.6+

On This Page

[Overview](#) ⓘ

[Symbols](#) ⓘ

[Relationships](#) ⓘ

Overview

Some of the methods in this protocol, such as

`tableView:objectValueForTableColumn:row:` and `numberOfRowsInTableView:` along with other methods that return data, are called frequently, so they must be efficient.

Note

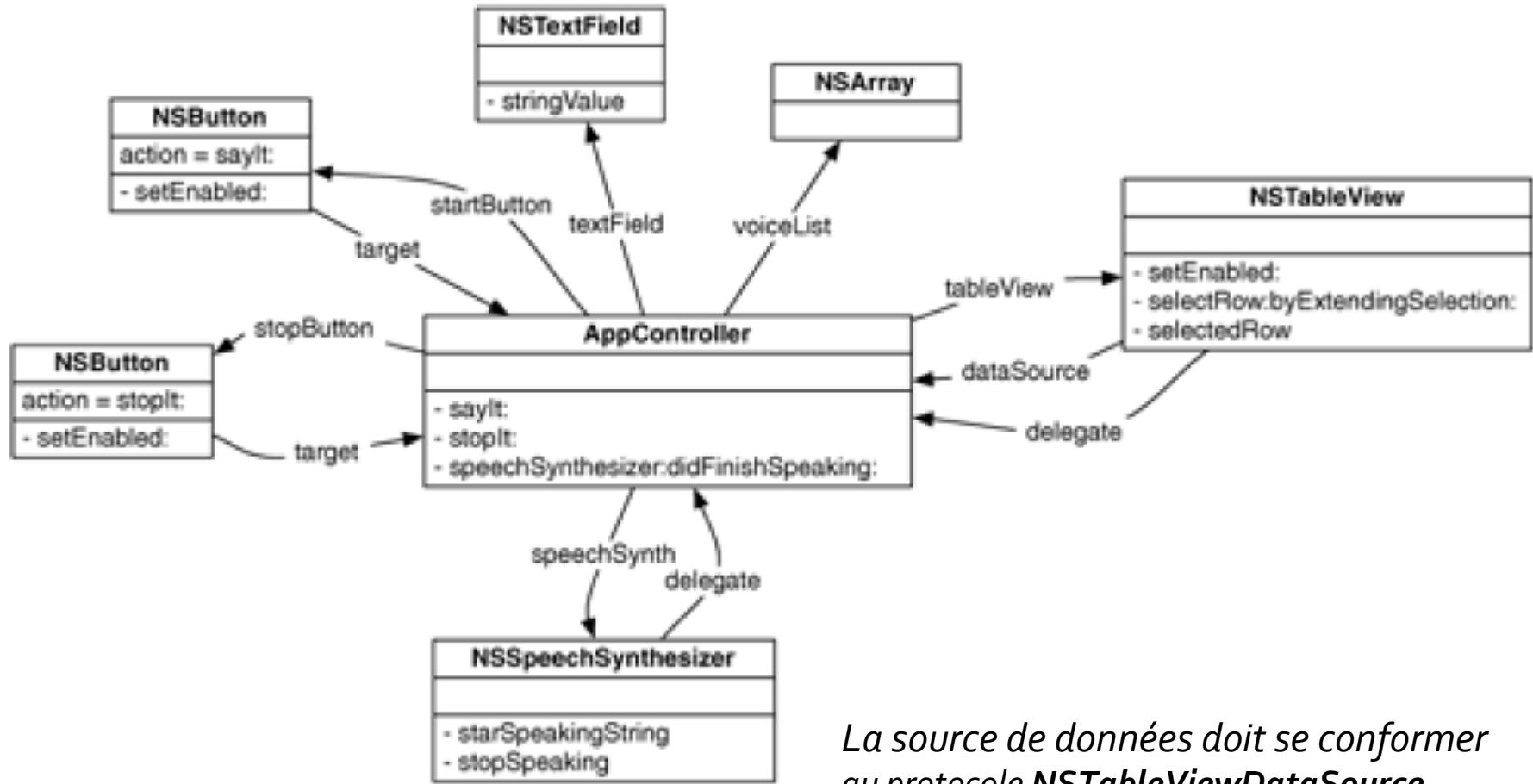
View-based table views must not use the `tableView:setObjectValue:forTableColumn:row:` method for setting values. Instead the views must explicitly set the values for the fields, or use Cocoa bindings. Likewise, use target/action for editing. See [Table View Programming Guide for Mac](#) for more information on populating view-based and cell-based table views.

If you're not using Cocoa bindings to provide data to the table view, the following methods are required:

- `numberOfRowsInTableView:`
- `tableView:objectValueForTableColumn:row:`
- `tableView:setObjectValue:forTableColumn:row:` (cell-based tables only)

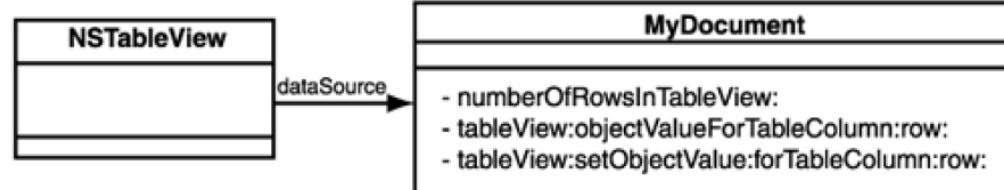
To learn more about Cocoa bindings, see [Cocoa Bindings Programming Topics](#).

■ NSTableView, sa source de données et son délégué

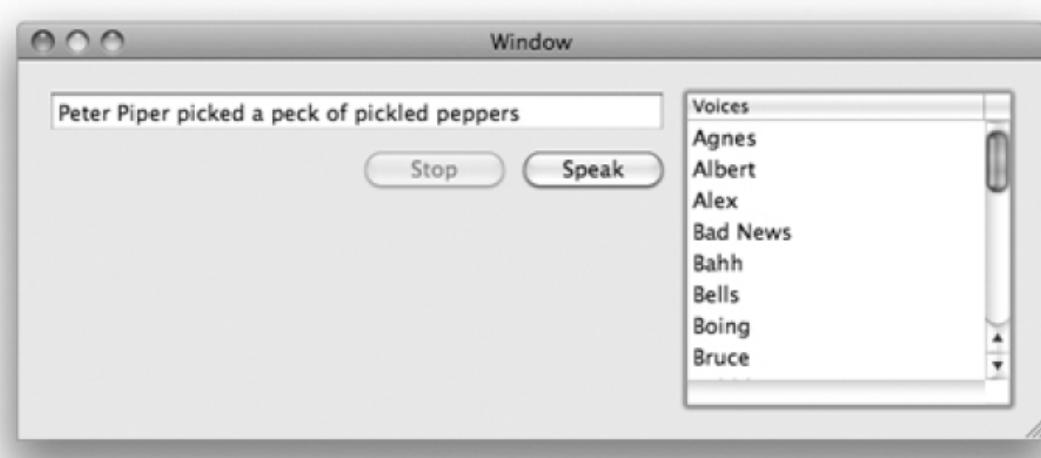


*La source de données doit se conformer
au protocole **NSTableViewDataSource**
Doit implémenter les deux méthodes :*

*protocole délégué
NSTableDelegate*



III *NSTableView, sa source de données et son délégué*



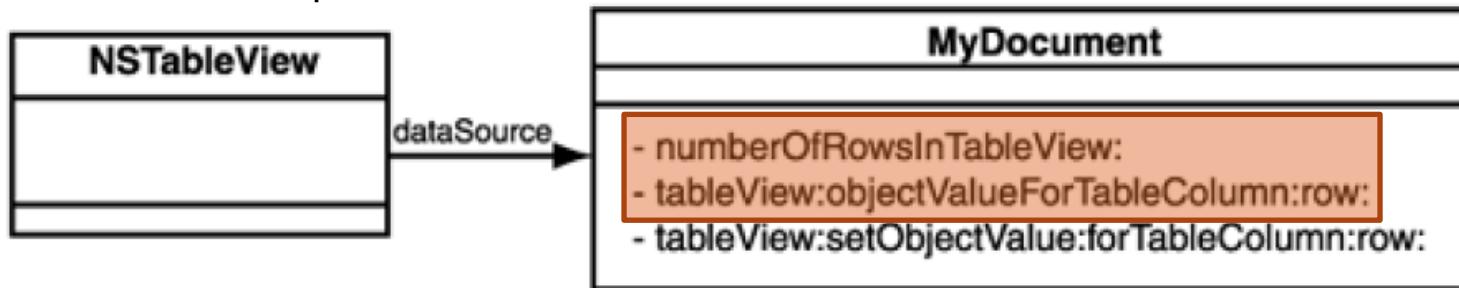
```
#import <Cocoa/Cocoa.h>

@interface AppController : NSObject
{
    IBOutlet NSTextField *textField;
    IBOutlet NSButton *startButton;
    IBOutlet NSButton *stopButton;
    IBOutlet NSTableView *tableView;
    NSArray *voiceList;
    NSSpeechSynthesizer *speechSynth;
}
```

```
- (id)init
{
    [super init];
    speechSynth = [[NSSpeechSynthesizer alloc] initWithVoice:nil];
    [speechSynth setDelegate:self];
    voiceList = [[NSSpeechSynthesizer availableVoices] retain];
    return self;
}
```

NSTableView, sa source de données et son délégué

Les méthodes pour la source de données



```
- (int)numberOfRowsInTableView: (NSTableView *) tv
{
    return [voiceList count];
}

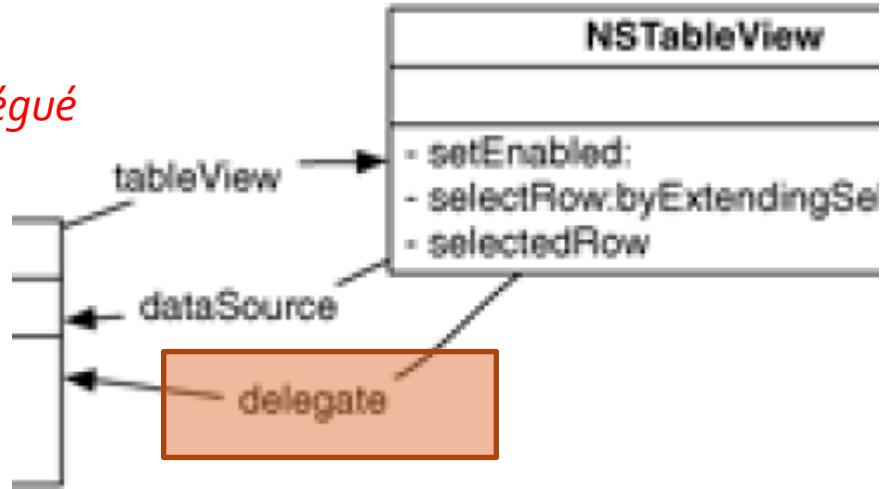
- (id)tableView: (NSTableView *) tv
    objectValueForTableColumn: (NSTableColumn *) tableColumn
    row: (int) row
{
    NSString *v = [voiceList objectAtIndex:row];
    return v;
}
```

```
NSString *v=[voiceList objectAtIndex:row];
NSDictionary *dict=[NSSpeechSynthesizer attributesForVoice:v];
return [dict objectForKey:NSVoiceName];
```



NSTableView, sa source de données et son délégué

Le délégué est AppController
Il est informé du
changement de sélection



```
- (void)tableViewSelectionDidChange:(NSNotification *)notification
{
    int row = [tableView selectedRow];
    if (row == -1) {
        return;
    }
    NSString *selectedVoice = [voiceList objectAtIndex:row];
    [speechSynth setVoice:selectedVoice];
    NSLog(@"new voice = %@", selectedVoice);
}
```



NSTableView, sa source de données et son délégué

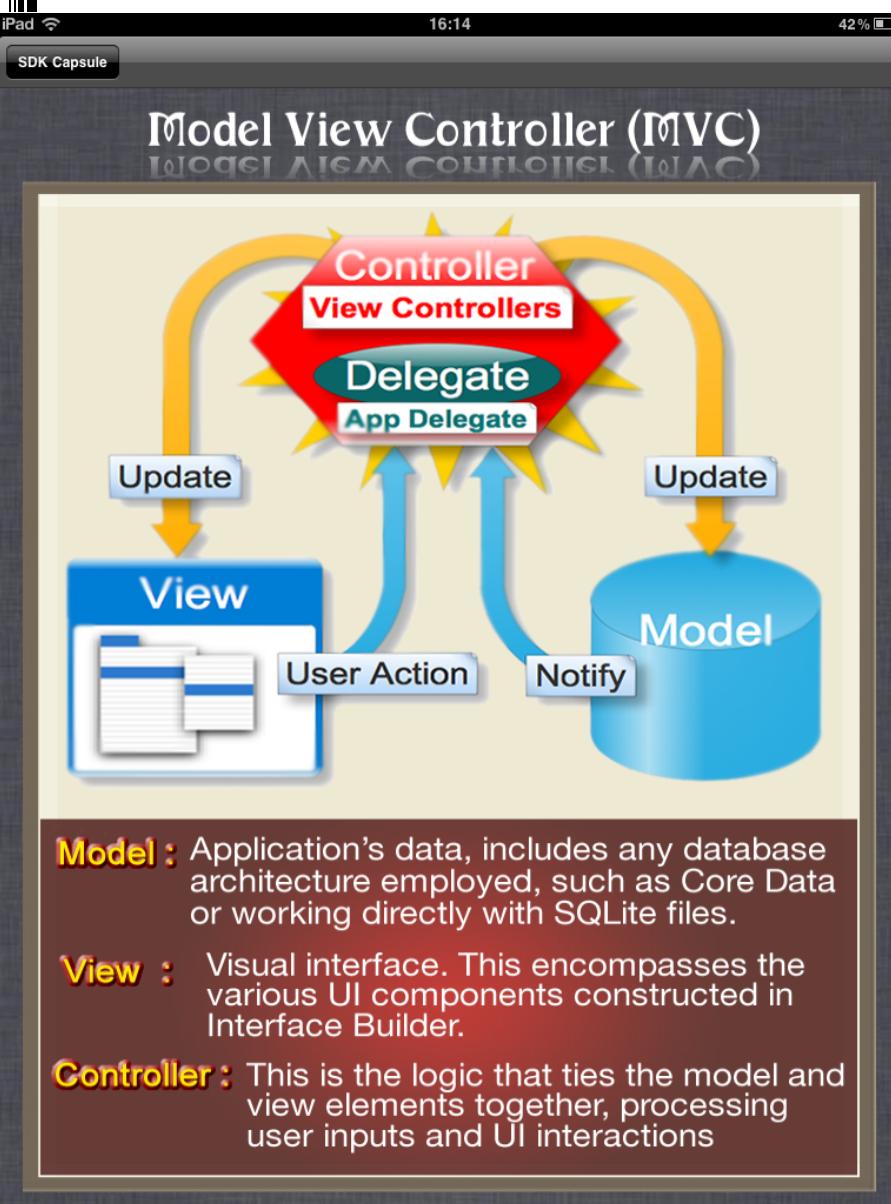
Gestion de la table dans AppController.m

```
- (IBAction)sayIt:(id)sender
{
    NSString *string = [textField stringValue];
    if ([string length] == 0) {
        return;
    }

    [speechSynth startSpeakingString:string];
    NSLog(@"Have started to say: %@", string);
    [stopButtonsetEnabled:YES];
    [startButtonsetEnabled:NO];
    [tableViewsetEnabled:NO];                                ←
}

- (void)speechSynthesizer:(NSSpeechSynthesizer *)sender
    didFinishSpeaking:(BOOL)complete
{
    NSLog(@"complete = %d", complete);
    [stopButtonsetEnabled:NO];
    [startButtonsetEnabled:YES];
    [tableViewsetEnabled:YES];                                ←
}
```

Modèle MVC : structure générale de l'application



Le modèle : c'est l'ensemble des classes qui gèrent les données de l'application

Objets métiers

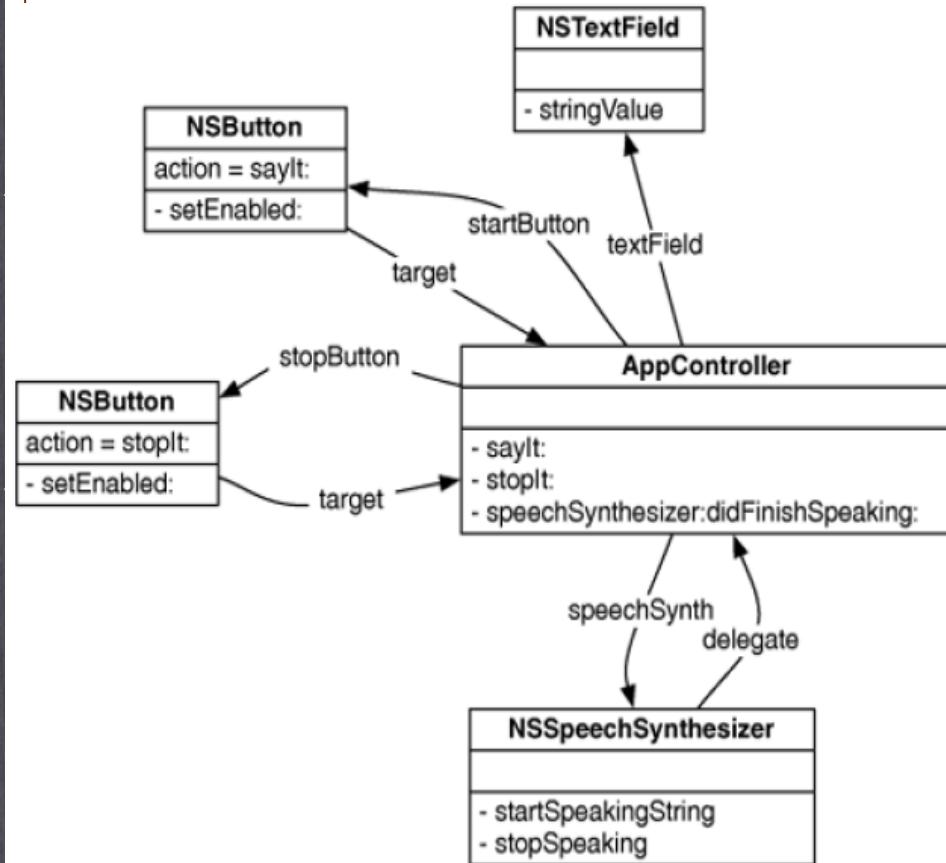
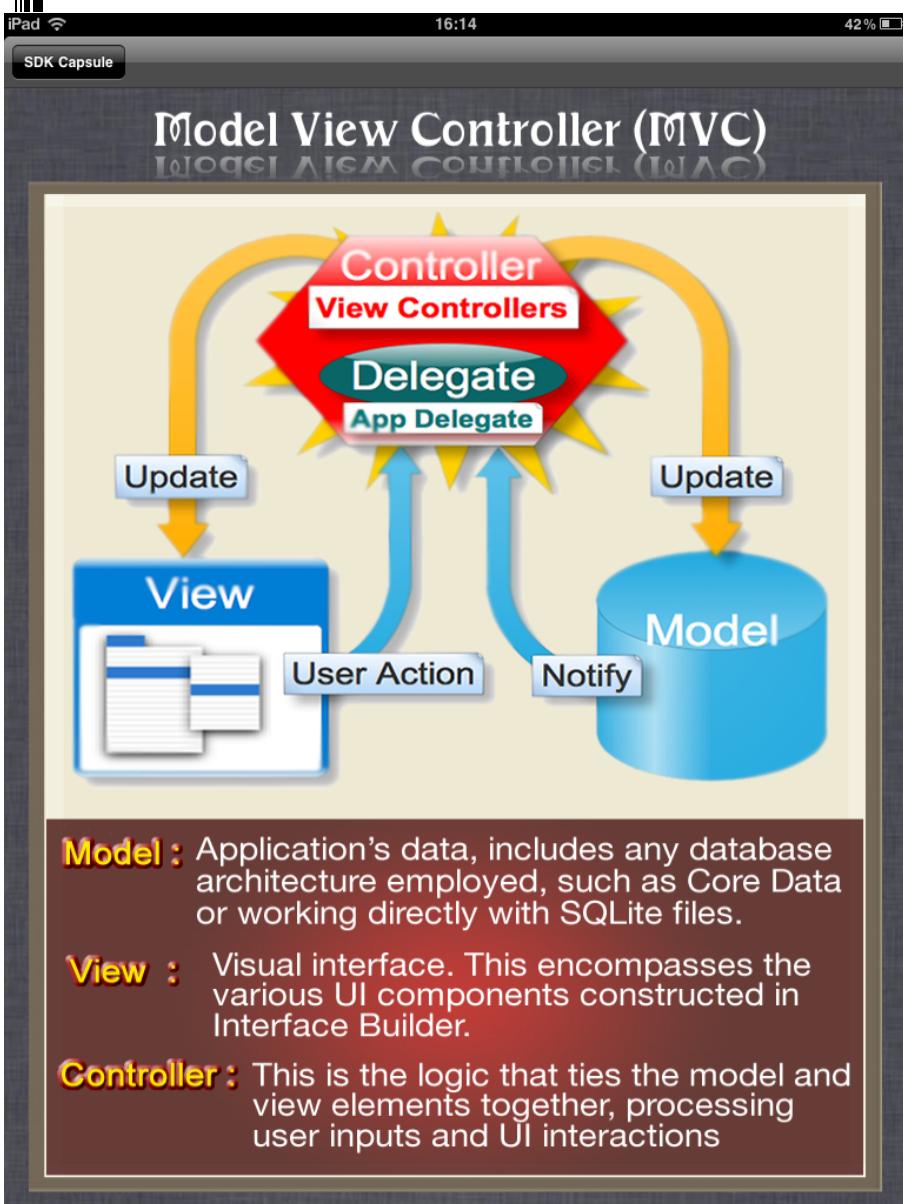
La vue : composée de la hiérarchie des vues et de la fenêtre de l'application

Les contrôleurs : qui comportent les contrôleurs de vues et l'application

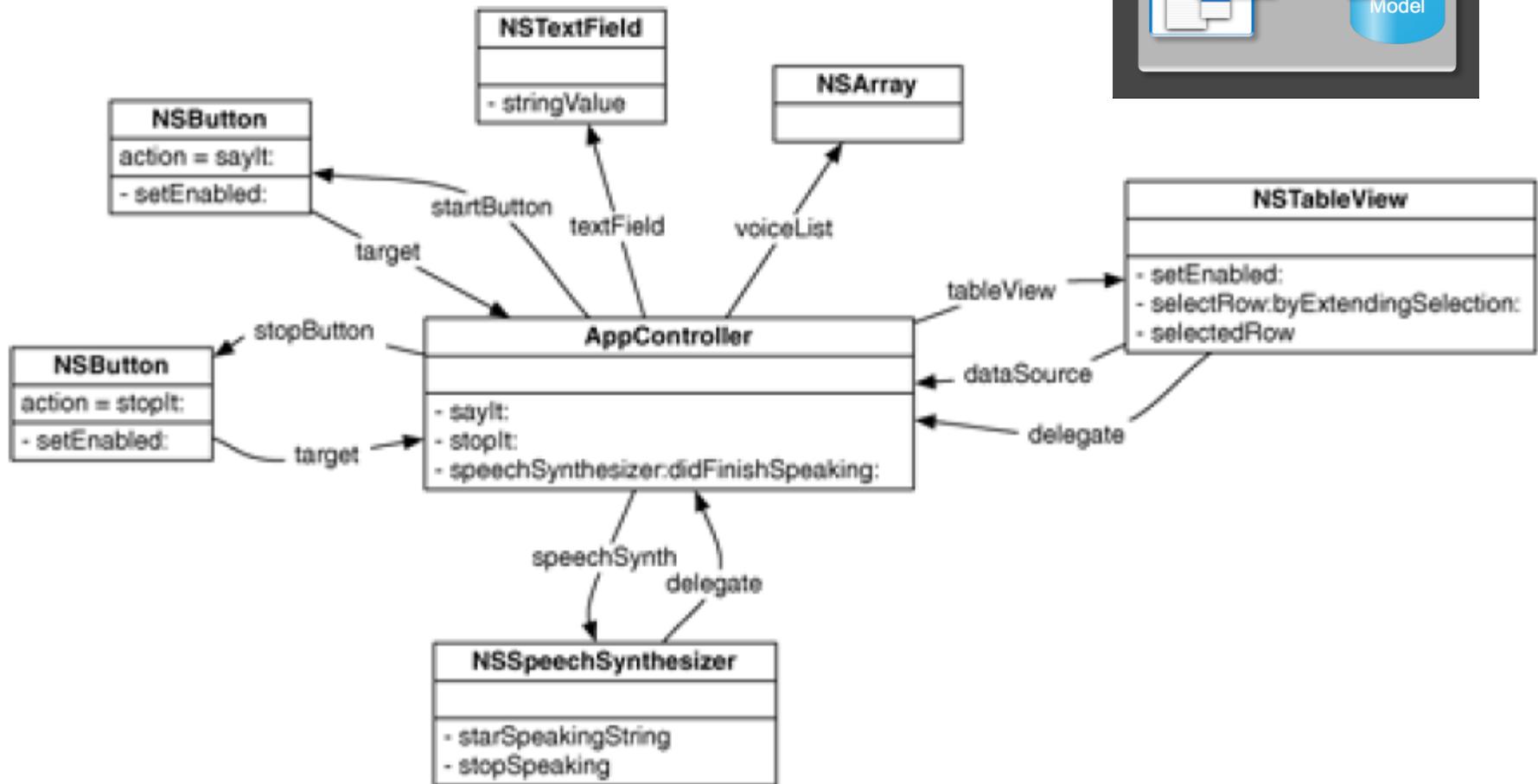
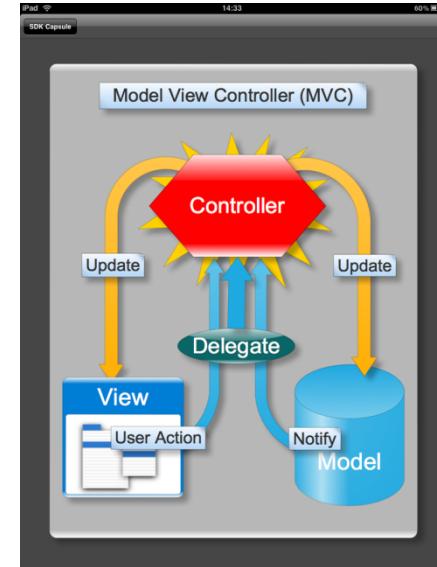
La délégation permet de modifier le comportement d'un objet sans avoir à le modifier ni le dériver.

L'objet délégué reçoit un message avant ou après chaque opération effectuée par son propriétaire

Modèle MVC : structure générale de l'application



Modèle MVC : structure générale de l'application



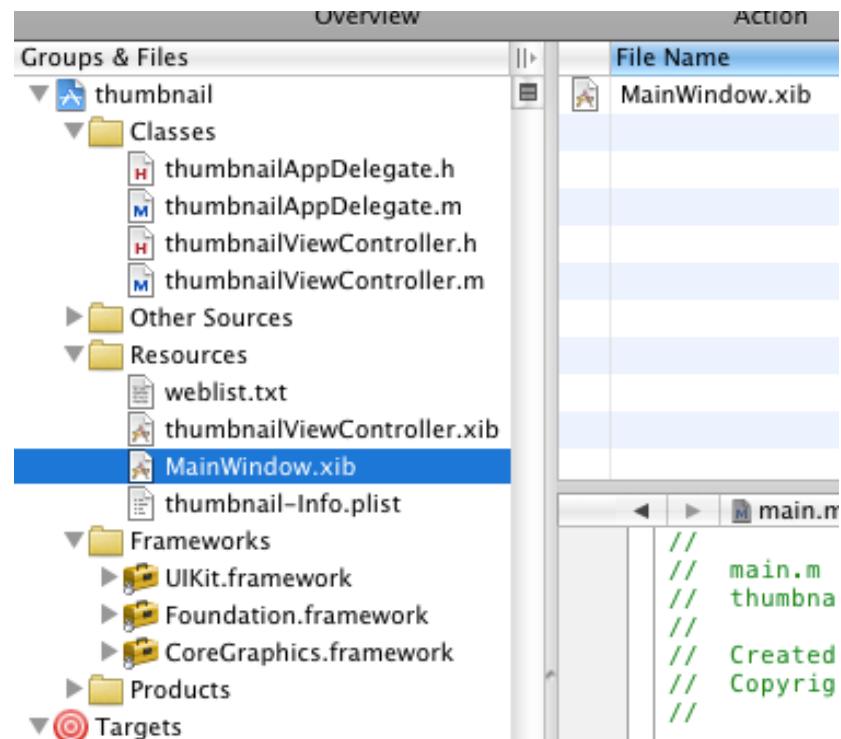
III Structurer une application

L'application charge un fichier NIB (MainWindow.xib) qui contient :

- une fenêtre et éventuellement des vues
- un délégué d'application : AppDelegate
- un ou plusieurs contrôleurs de vue

Chaque contrôleur de vue possède lui-même un fichier nib qui contient :

- Une hiérarchie de vues à l'intérieur de sa vue principale
- À son tour, un ou plusieurs contrôleurs de vue
- Éventuellement d'autres objets.



III Structurer une application

Toute application contient une fonction **main** appelée lors de son lancement

- Création d'un pool d'autolibération principal
- Appel de la fonction UIApplicationMain

- Libération du pool
- Fin de l'application



```
// main.m
// thumbnail
//
// Created by mmenard on 06/11/10.
// Copyright 2010 __MyCompanyName__. All rights reserved.
//

#import <UIKit/UIKit.h>

int main(int argc, char *argv[]) {
    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];
    int retVal = UIApplicationMain(argc, argv, nil, nil);
    [pool release];
    return retVal;
}
```

- création d'un objet de la classe *UIApplication*
- chargement du fichier *MainWindow.xib*
(le propriétaire est *UIApplication*)
- création de la boucle événement
- Lancement de la boucle d'événement tant que l'application n'a pas été quittée
- Fonction terminée

Le fichier MainWindow.xib est chargé au lancement de l'application



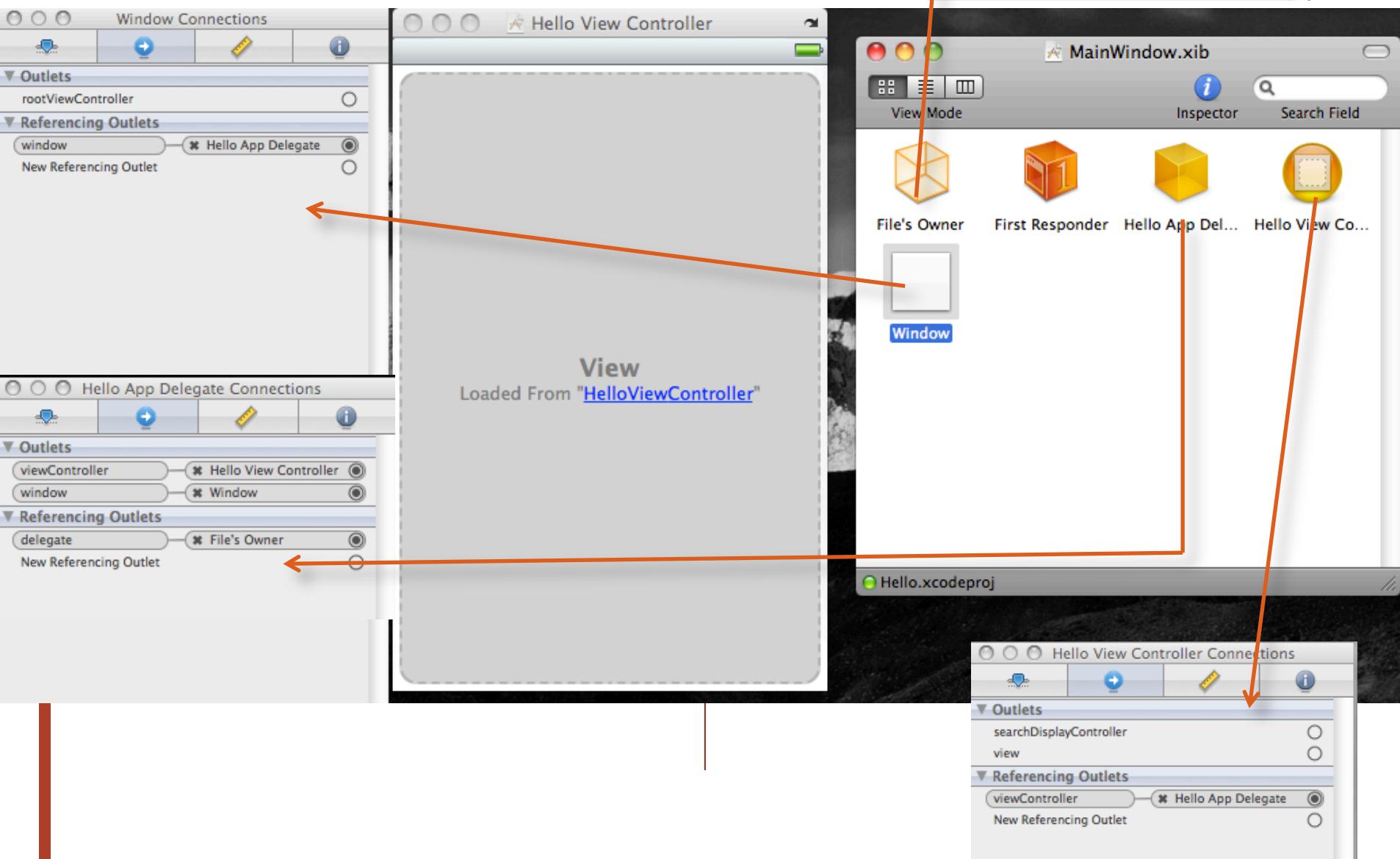
Le propriétaire est de la classe UIApplication

Représente la vue de l'interface en cours d'édition par l'utilisateur

Le délégué d'application

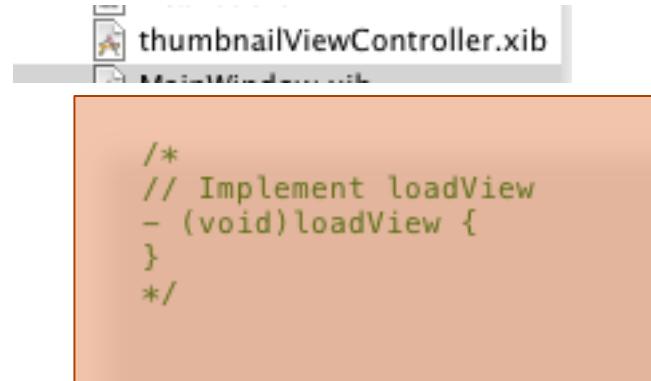
*De type UIWindow.
Cet objet est la fenêtre de l'application dans laquelle toutes les vues seront affichées*

Le délégué d'application prend la main



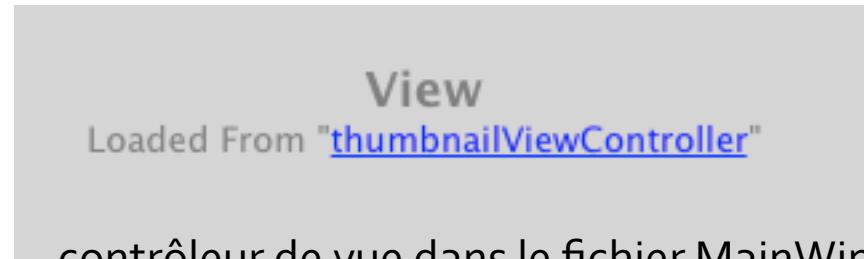
Contrôleur de vue

MyApplicationViewController



Le chargement du fichier NIB MainWindow.xib crée un objet MyApplicationViewController associé au fichier NIB MyApplicationViewController.xib

C'est l'utilisation de la propriété view du contrôleur de vue qui provoque le chargement du fichier nib associé

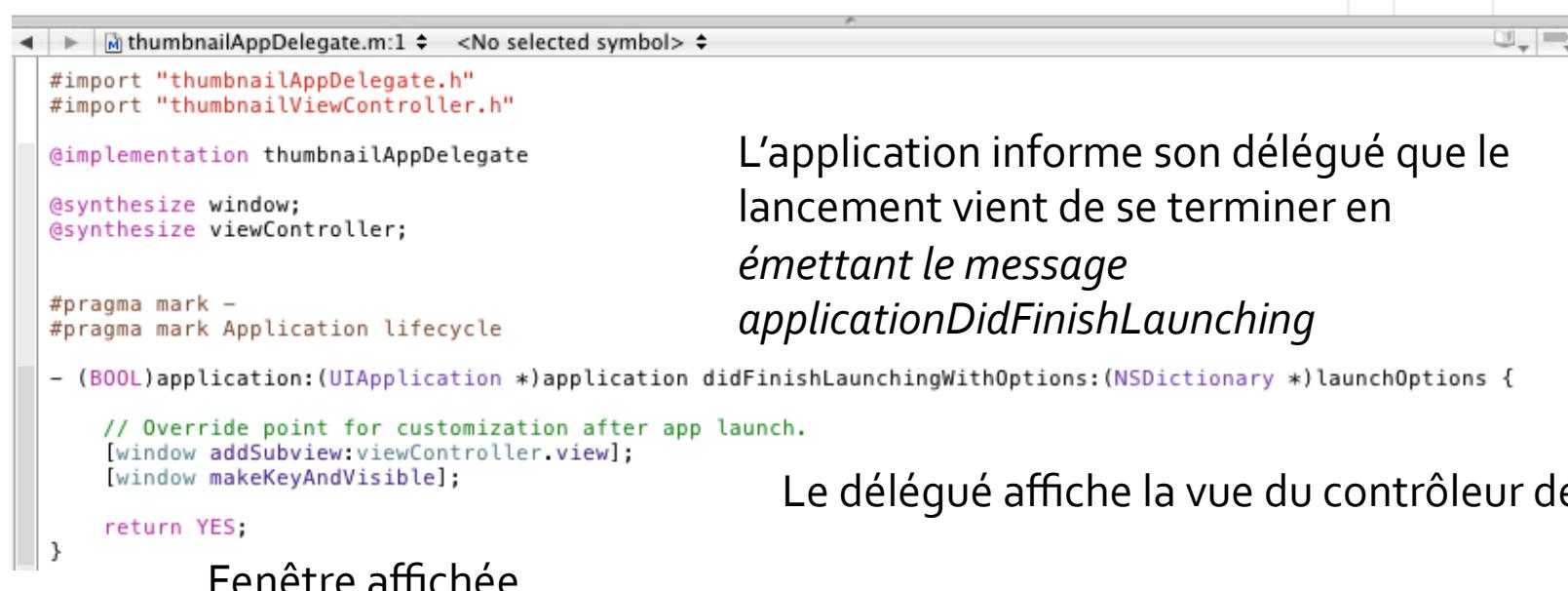


contrôleur de vue dans le fichier MainWindow.xib

Délégué d'application

Fonction *UIApplicationMain* :

- > Création d'une instance de UIApplication
- > Création du délégué d'application
 - > en charge d'associer la vue du contrôleur à la vue de window



```
#import "thumbnailAppDelegate.h"
#import "thumbnailViewController.h"

@implementation thumbnailAppDelegate
@synthesize window;
@synthesize viewController;

#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after app launch.
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];
    return YES;
}
```

L'application informe son délégué que le lancement vient de se terminer en émettant le message *applicationDidFinishLaunching*

Le délégué affiche la vue du contrôleur de vue

Fenêtre affichée

Contrôleurs de vue Et storyboard

mapLP

- mapLP
 - googlemarkers.json
 - FliteTTS.m
 - FliteTTS.h
 - flite-1.4-iphone
 - AppDelegate.h
 - AppDelegate.m
- Main.storyboard
- FirstViewController.h
- FirstViewController.m
- SecondViewController.h
- SecondViewController.m

First Scene

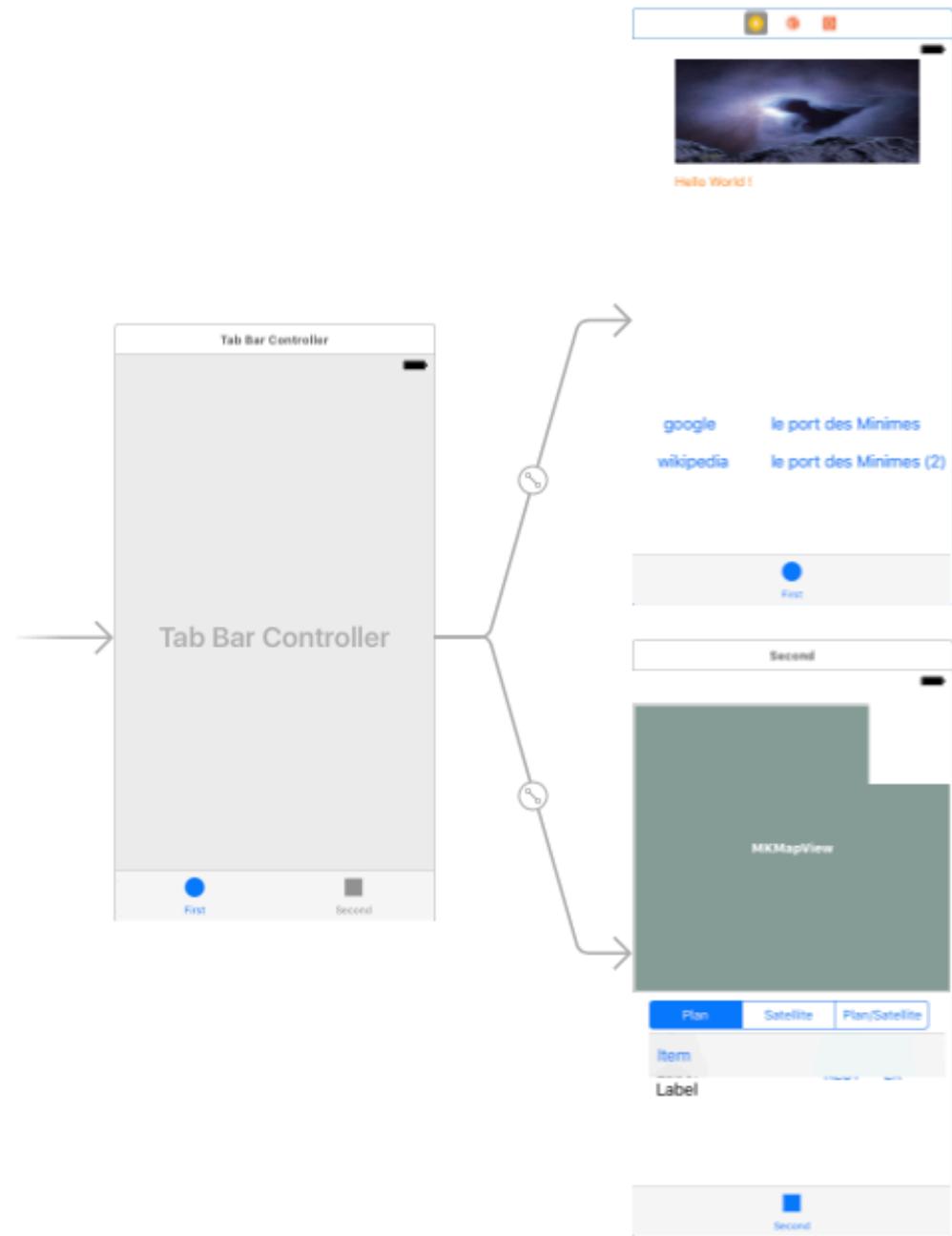
- First
- First Responder
- Exit

Second Scene

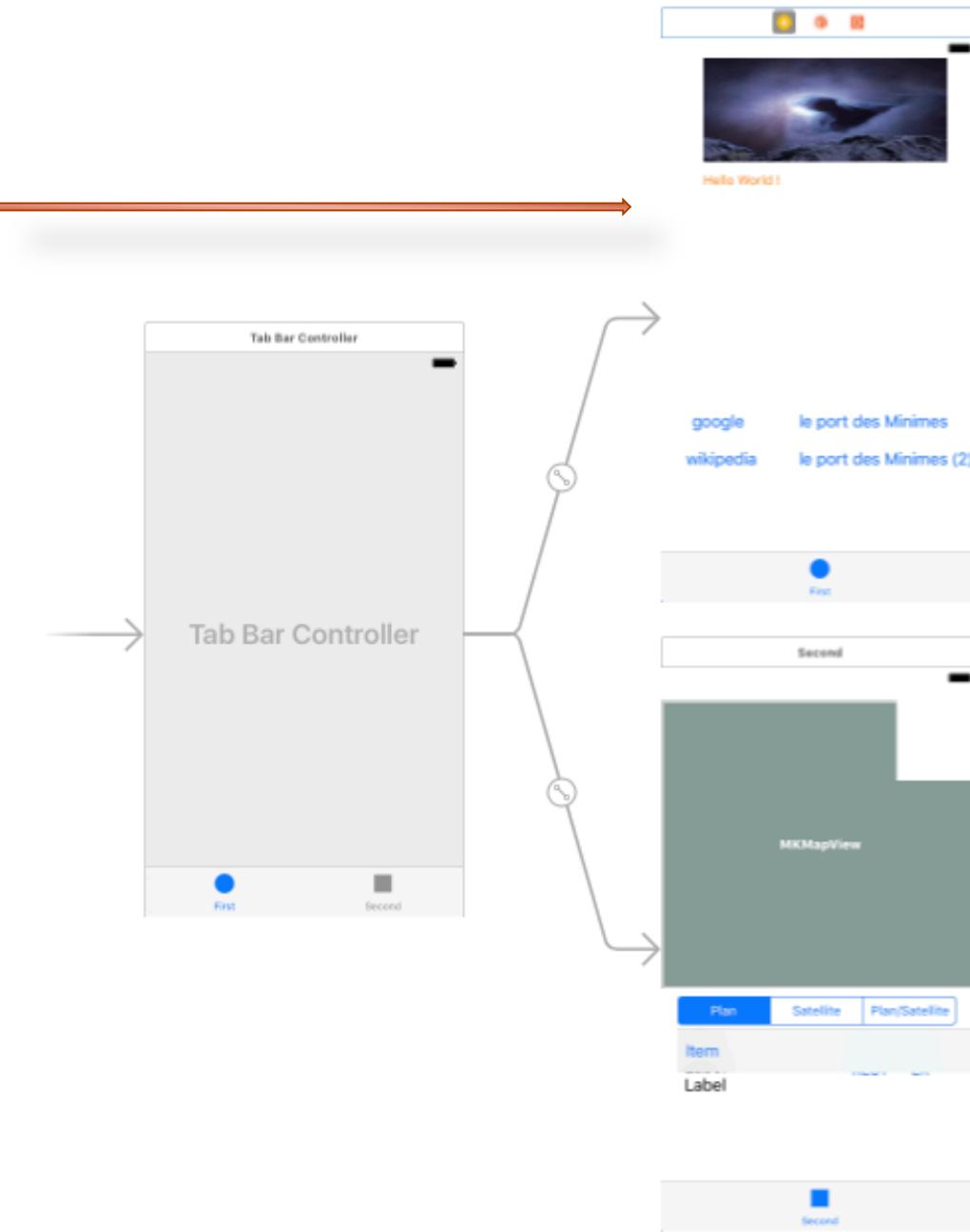
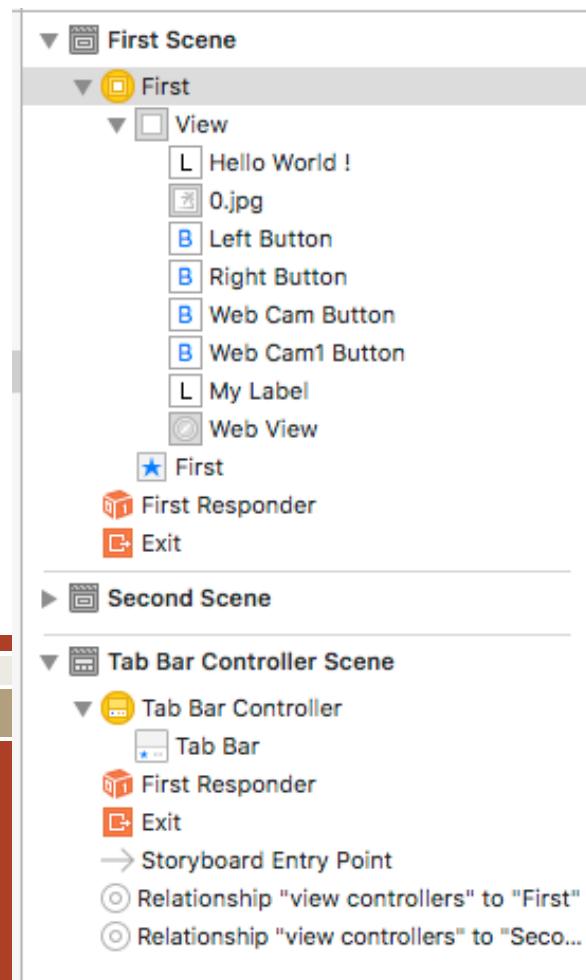
- Second
- View
- Second
- First Responder
- Exit

Tab Bar Controller Scene

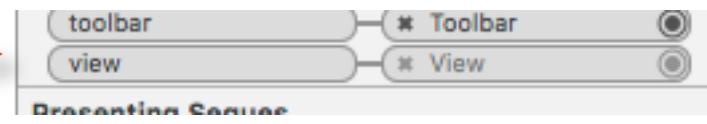
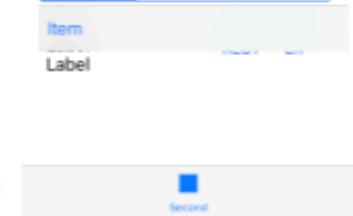
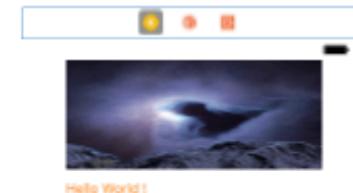
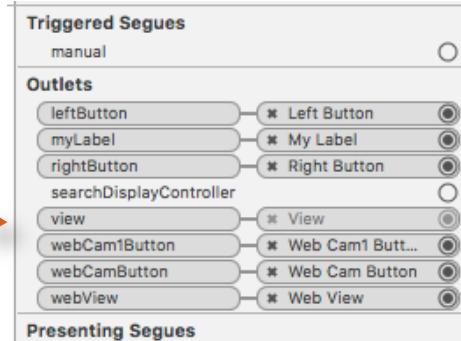
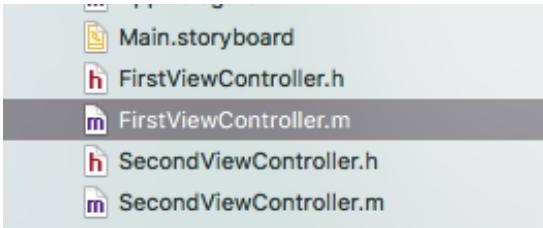
- Tab Bar Controller
 - Tab Bar
 - First Responder
 - Exit
- Storyboard Entry Point
- Relationship "view controllers" to "First"
- Relationship "view controllers" to "Seco..."



Contrôleurs de vue Et storyboard



Contrôleurs de vue Et storyboard



☰ Les notifications : NSNotificationCenter

Exemples

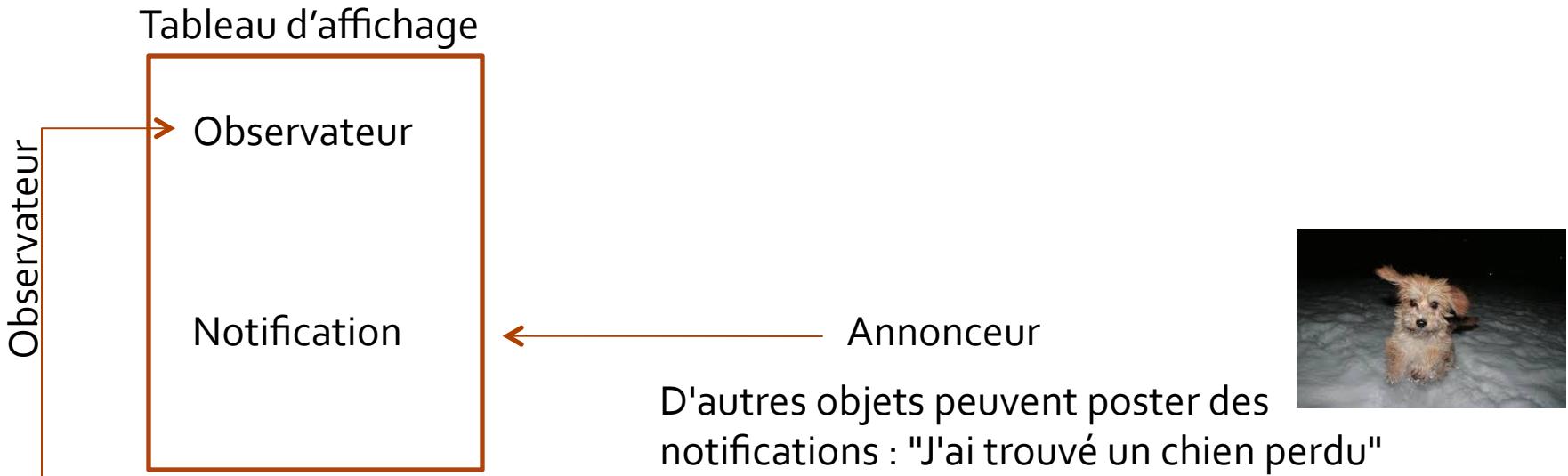
1. Les fenêtres signalent qu'elles ont changé de taille
2. Lorsque la sélection d'une vue tableau change, la vue poste une notification
3. Des documents à l'écran doivent changer de couleur d'arrière-plan.
 - On inscrit tous les objets MyDocument en tant qu'observateur.
 - Le contrôleur de préférences émettra une notification lorsque l'utilisateur choisira une nouvelle couleur.
 - En voyant la notification, les objets MyDocument modifieront la couleur d'arrière plan



Les notifications ne franchissent pas les limites de l'application



Les notifications : NSNotificationCenter



Les objets s'inscrivent sur le tableau d'affichage pour indiquer qu'ils sont intéressés par certains avis : "Merci de m'écrire si quelqu'un trouve un chien"

PERDU



Disparu dans la nuit du 10 Février
à La Roque d'Anthéron

TRES GENTIL !! mais CRAINTIF !!
(ne mord pas)

Tél : 06.62.31.84.16
Ou 06.03.35.32.80

La notification est transmise à tous les objets qui ont signalé leur intérêt pour cet avis

Les notifications : NSNotificationCenter

Les objets

Une notification possède deux variables d'instance importantes : name et object et deux méthodes intéressantes :

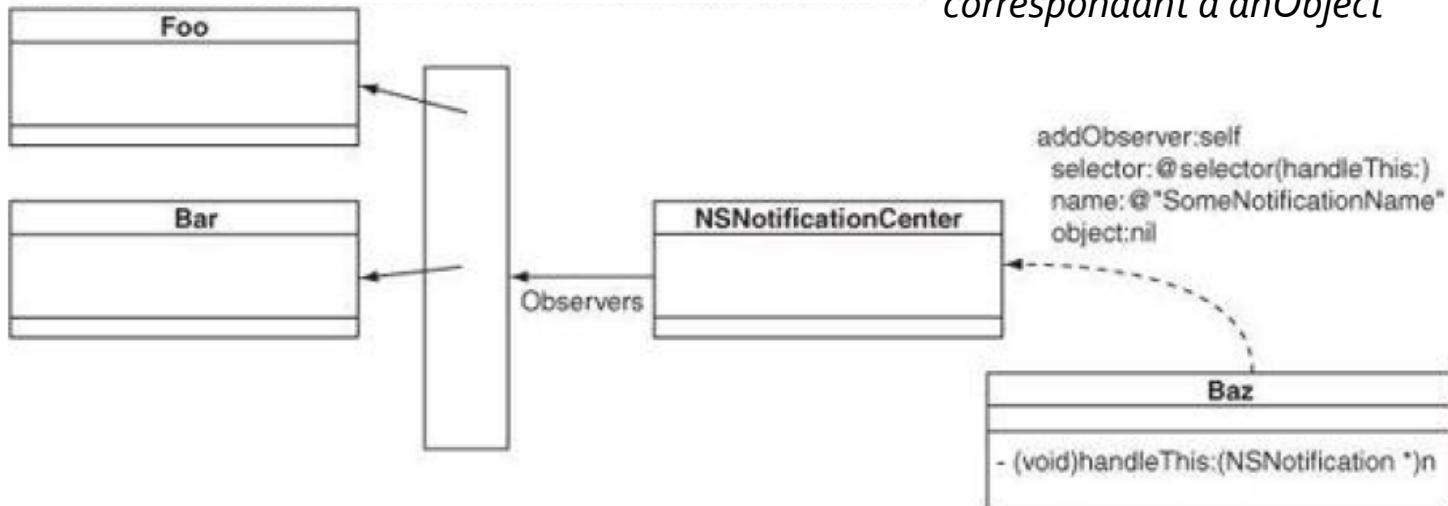
- (NSString *) name
- (id) object → pointeur vers l'objet qui émet la notification

NSNotificationCenter permet d'enregister des objets observateurs, de poster des notifications, et de désinscrire des observateurs

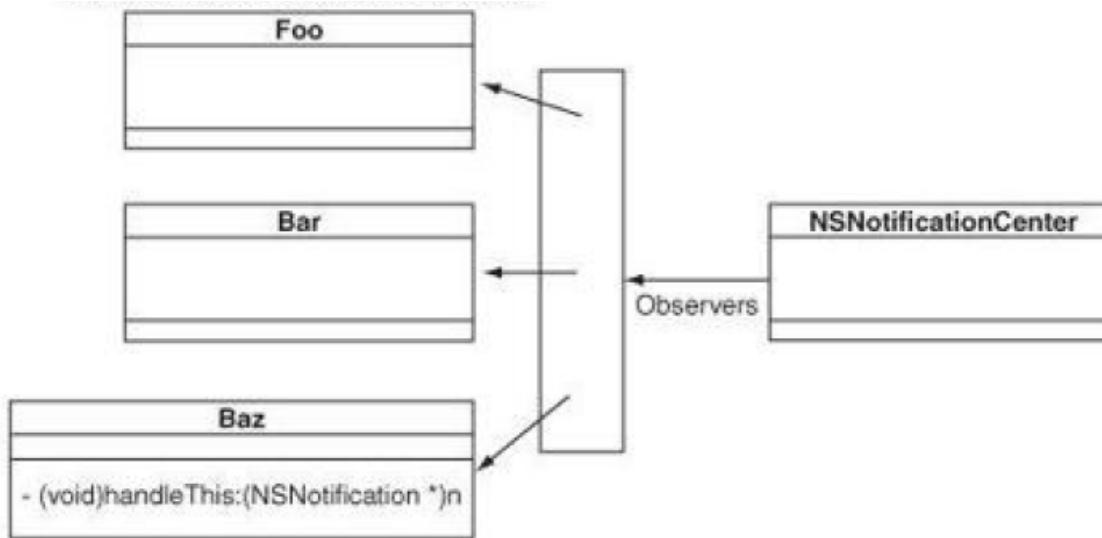
```
+ (NSNotificationCenter *)defaultCenter  
  
- (void)addObserver:(id)anObserver  
selector:(SEL)aSelector  
name:(NSString *)notificationName  
object:(id)anObject
```

→ Inscrit **anObserver** pour qu'il reçoive les notifications de nom **notificationName** et contenant **anObject**. Lorsque une notification nommée **notificationName** et contenant l'objet **anObject** est postée, **anObserver** reçoit un message **aSelector** avec la notification en argument

Les notifications :NSNotificationCenter



Inscrit **anObserver** Baz pour qu'il recoive des notifications nommées SomeNotificationName



III Les notifications : NSNotificationCenter

Si notificationName est nil :

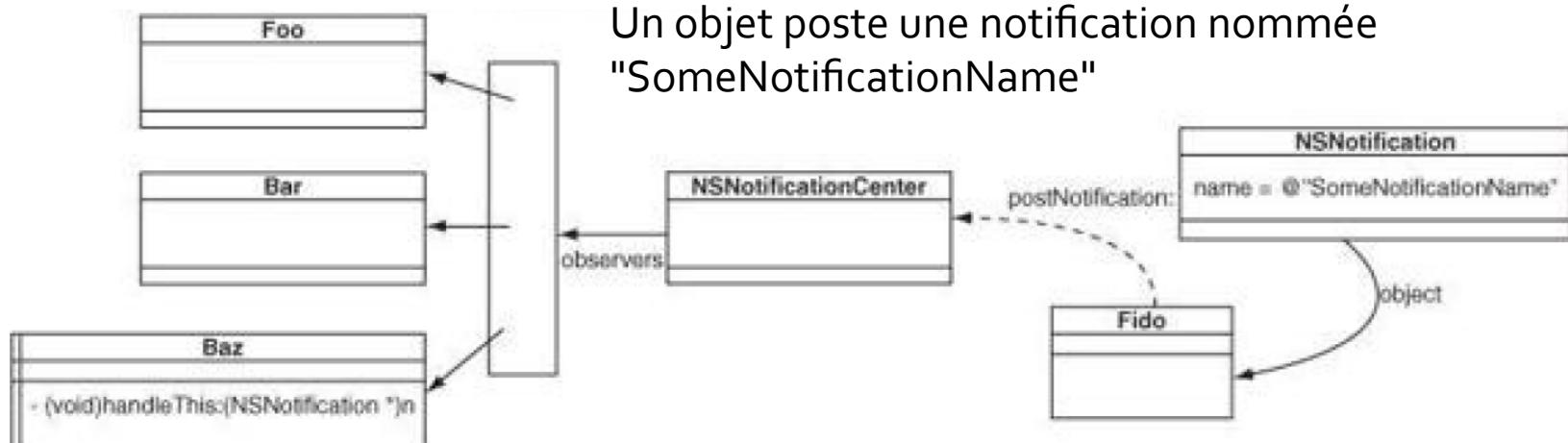
le centre envoie à l'observateur toutes les notifications qui contiennent un objet correspondant à anObject

Si anObject est nil :

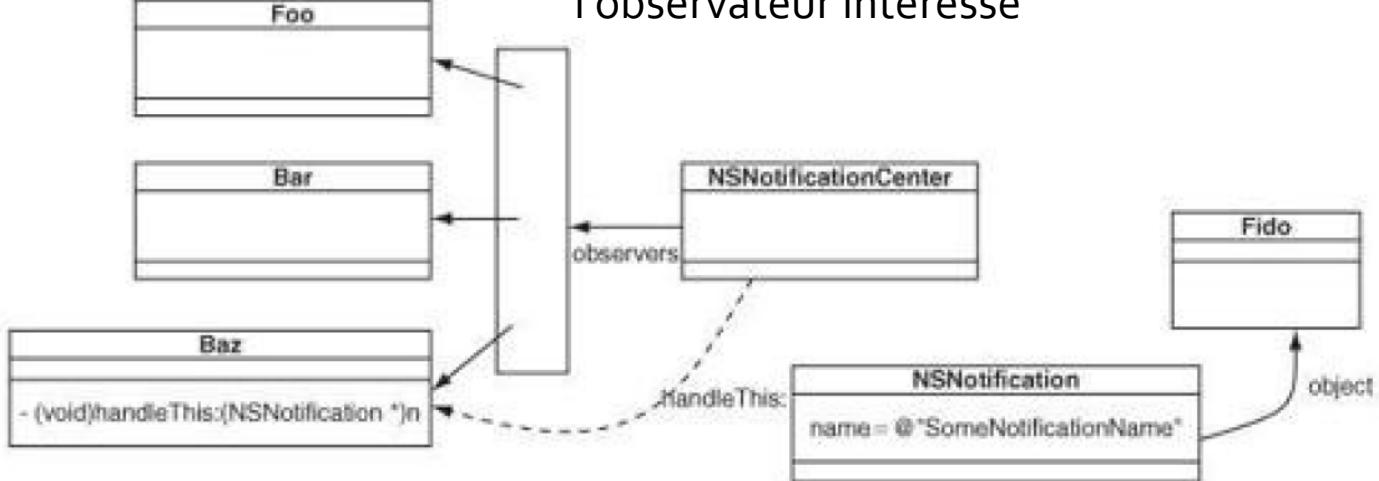
le centre envoie à l'observateur toutes les notifications nommées notificationName

- (void)postNotification:(NSNotification *)notification
- (void)postNotificationName:(NSString *)aName object:(id)anObject
- (void)removeObserver:(id)observer

Les notifications : NSNotificationCenter



Un objet poste une notification nommée "SomeNotificationName"



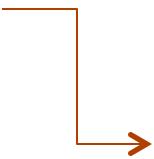
Les notifications : NSNotificationCenter

Exemple de code : poster une notification
PreferenceController.m

```
- (IBAction)changeBackgroundColor:(id)sender
{
    NSColor *color = [colorWell color];
    [PreferenceController setPreferenceTableBgColor:color];
```

```
NSNotificationCenter *nc = [NSNotificationCenter defaultCenter];
NSLog(@"Sending notification");
[nc postNotificationName:BNRColorChangedNotification object:self];
}
```

*La notification se nommera
@"BNRColorChanged"*



```
extern NSString * const
BNRColorChangedNotification =
@""BNRColorChanged";
(variable globale)
```

Les notifications : NSNotificationCenter

Exemple de code : inscrire un observateur
Dans le fichier MyDocument.m

```
- (id)init
{
    self = [super init];
    if (self) {
        employees = [[NSMutableArray alloc] init];
        NSNotificationCenter *nc = [NSNotificationCenter defaultCenter];
        [nc addObserver:self
                    selector:@selector(handleColorChange:)
                    name:BNRColorChangedNotification
                    object:nil];

        NSLog(@"%@", @"Registered with notification center");
    }
    return self;
}

[[NSNotificationCenter defaultCenter] removeObserver:self];
```

Éléments importants : objet qui constitue l'observateur, les noms des notifications qui l'intéressent, et le message envoyé lors de l'arrivée d'une notification
Nous pouvons préciser que seules les notifications qui contiennent un certain objet intéressent l'observateur

☰ Les notifications : NSNotificationCenter

Exemple de code : traiter la notification
Dans le fichier MyDocument.m

Lorsque la notification arrive, la méthode handleColorChange: est invoquée

```
- (void)handleColorChange:(NSNotification *)note
{
    NSLog(@"%@", @"Received notification: %@", note);
}
```

Les notifications : NSNotificationCenter

Exemple de code le dictionnaire userInfo
Dans le fichier PreferenceController.m

```
- (IBAction)changeBackgroundColor:(id)sender
{
    NSColor *color = [sender color];
    [PreferenceController setPreferenceTableBgColor:color];
    NSNotificationCenter *nc = [NSNotificationCenter defaultCenter];
    NSLog(@"Sending notification");
    NSDictionary *d = [NSDictionary dictionaryWithObject:color forKey:@"color"];
    [nc postNotificationName:BNRColorChangedNotification object:self userInfo:d];
}
```

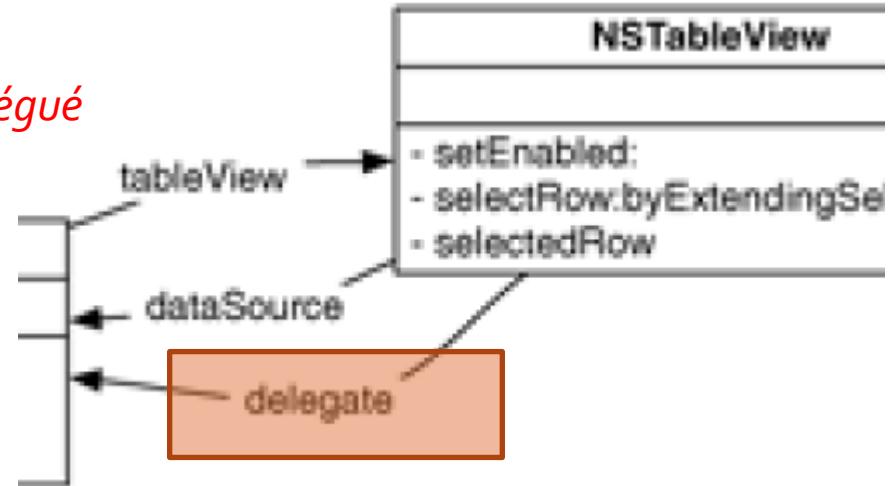
Chaque notification possède une variable nommée userInfo à laquelle peut être attaché un NSDictionary qui contient les autres informations à passer aux observateurs

Dans MyDocument.m nous lisons la couleur à partir du dictionnaire userInfo

```
- (void)handleColorChange:(NSNotification *)note
{
    NSLog(@"Received notification: %@", note);
    NSColor *color = [[note userInfo] objectForKey:@"color"];
    [tableView setBackgroundColor:color];
}
```

NSTable View , sa source de données et son délégué

Le délégué est AppController
Il est informé du
changement de sélection



```
- (void)tableViewSelectionDidChange:(NSNotification *)notification
{
    int row = [tableView selectedRow];
    if (row == -1) {
        return;
    }
    NSString *selectedVoice = [voiceList objectAtIndex:row];
    [speechSynth setVoice:selectedVoice];
    NSLog(@"new voice = %@", selectedVoice);
}
```



Si un objet Cocoa standard possède un délégué et poste des notifications, le délégué est automatiquement inscrit comme observateur pour les méthodes implémentées par l'objet

*Convention de nommage (exemple):
Notification NSWindowDidResizeNotification
- (void)windowDidResize:(NSNotification *)aNotification*

NSTableView et notification

Global Variable

NSTableViewSelectionDidChangeNotification

Posted after an `NSTableView` object's selection changes.

Language

[Swift](#) | [Objective-C](#)

Declaration

```
NSNotificationName NSTableViewSelectionDidChangeNotification;
```

SDK

macOS 10.0+

On This Page

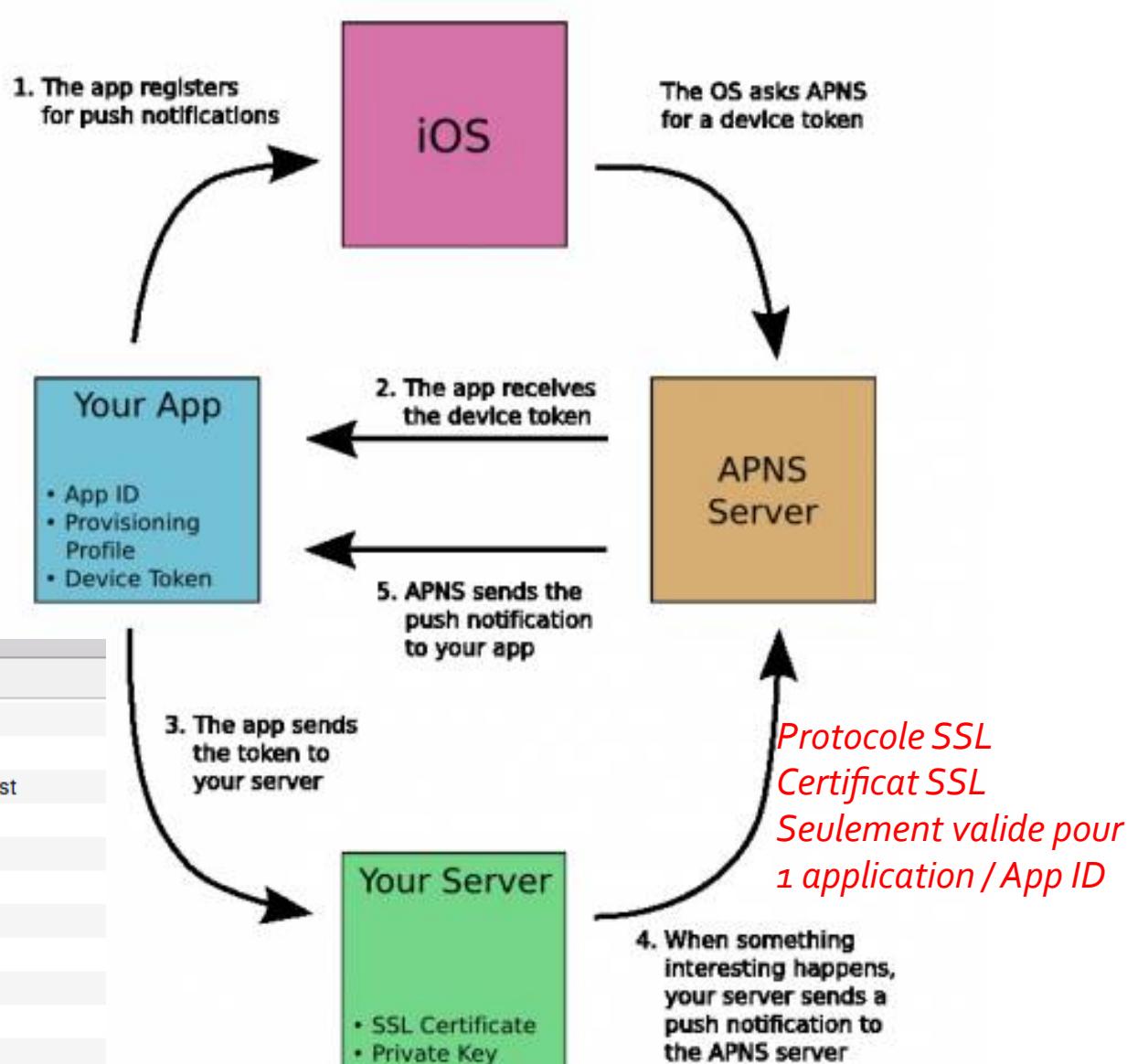
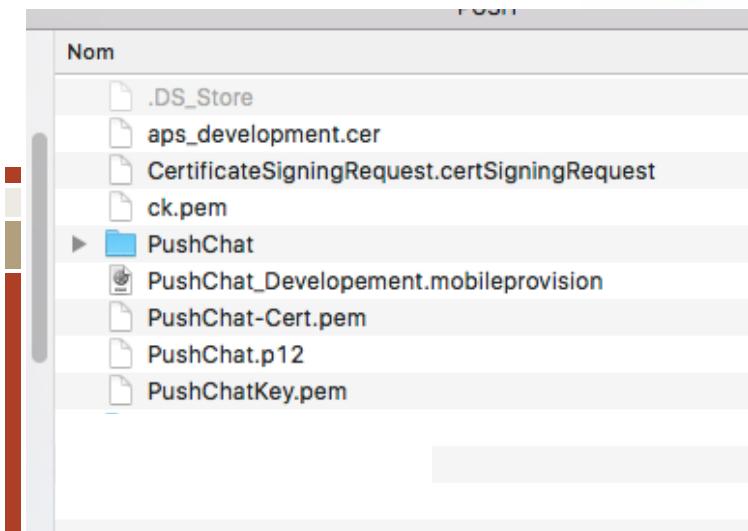
[Declaration](#) ⓘ

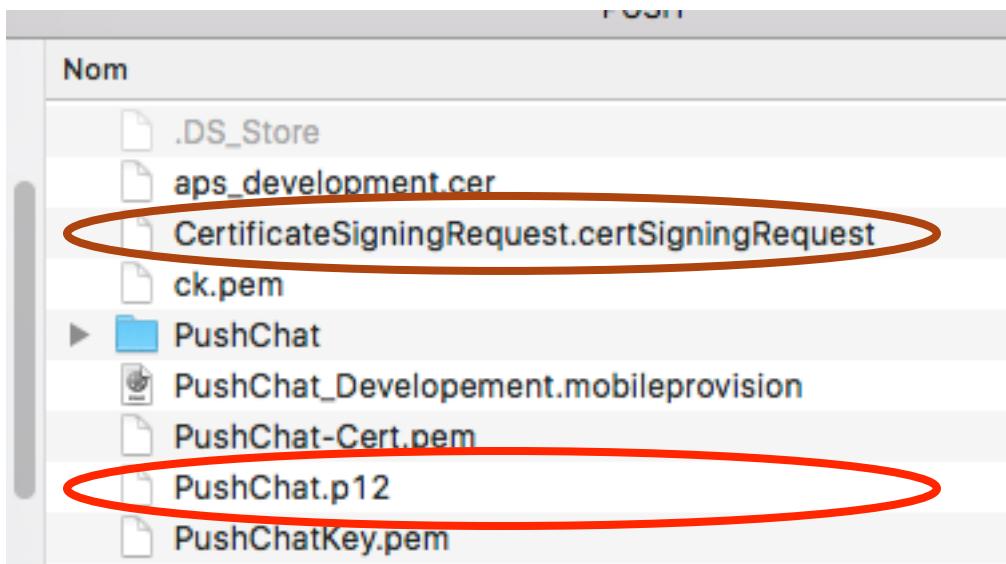
[Discussion](#) ⓘ

Discussion

The notification object is the table view whose selection changed. This notification does not contain a `userInfo` dictionary.

Les notifications push





Certificat

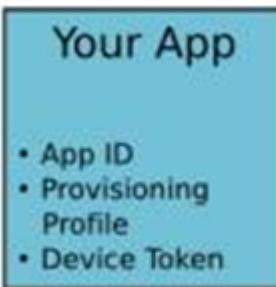
Requête de certificat (CSR / SSL)

Clé privée et clé publique

Profil de provisionnement

Clé publique, format PEM

Clé publique et privée, format p12





Trousseaux d'accès

Fichier

Édition

Présentation

Fenêtre

Aide

À propos de Trousseaux d'accès

Préférences... ⌘,

S.O.S. Trousseau ⌘⌘A

Assistant de certification ►

Visualiseur de ticket ⌘⌘K

Services ►

Masquer Trousseaux d'accès ⌘H

Masquer les autres ⌘⌘H

Tout afficher

Quitter Trousseaux d'accès ⌘Q

- [Ouvrir...](#)
- [Créer un certificat...](#)
- [Créer une autorité de certificat...](#)
- [Créer un certificat pour quelqu'un d'autre en tant qu'autorité de certificat...](#)
- [Demander un certificat à une autorité de certificat...](#)
- [Définir l'autorité de certificat par défaut...](#)
- [Évaluer « Apple Application Integration Certification Authority »...](#)

Assistant de certification

Informations sur le certificat

Saisissez les informations relatives au certificat que vous demandez. Cliquez sur Continuer pour demander un certificat

Adresse e-mail de l'utilisateur : michel.menard@univ-lr.fr

Nom commun : 1

Adresse e-mail de l'AC : Requis

La requête est :

- Envoyée à l'autorité de certif. par e-mail
- Enregistrée sur disque
- Me laisser indiquer les informations sur la bi-clé

Continuer

Cliquez pour verrouiller le trousseau « session ».

Trousseaux

- [session](#)
- [Microsoft_Intermediate_C](#)
- [Éléments locaux](#)
- [Système](#)
- [Racines du système](#)

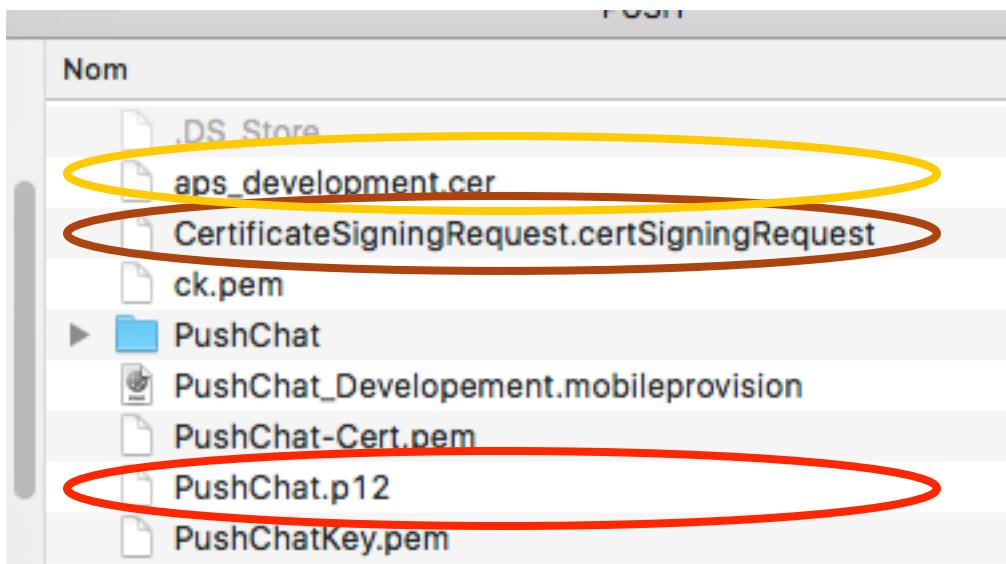
<key>

Type : clé publique, RSA, 2048 bits
Utilisation : Chiffrer, Dériver, Vérifier

Nom

	Type
iMessage Signing Key	clé publique
iMessage Signing Key	clé publique
iMessage Signing Key	clé publique
iMessage Signing Key	clé privée
iOS Developer: Michel Ménard (University of La Rochelle (Informatique))	clé publique
iOS Developer: Michel Ménard (University of La Rochelle (Informatique))	clé publique
iOS Developer: Michel Ménard (University of La Rochelle (Informatique))	clé privée
iOS Developer: Michel Ménard (University of La Rochelle (Informatique))	clé privée
Michel Menard	clé privée
Michel Menard	clé privée
mmenard	clé privée
mmenard	clé privée
PushChat	clé publique
PushChat	clé privée

export p12 ←



Certificat

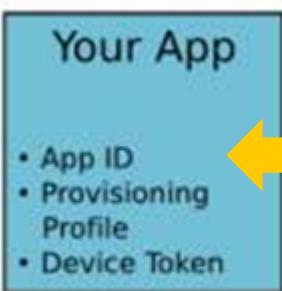
Requête de certificat (CSR / SSL)

Clé privée et clé publique

Profil de provisionnement

Clé publique, format PEM

Clé publique et privée, format p12



Certificates, Identifiers & Profiles

iOS Apps

Certificates

- All
- Pending
- Development
- Production

Identifiers

- App IDs
- Pass Type IDs
- Website Push IDs
- iCloud Containers
- App Groups

Devices

- All
- Apple TV

Register iOS App ID



ID

Registering an App ID

The App ID string contains two parts separated by a period (.)—an App ID Prefix that is defined as your Team ID by default and an App ID Suffix that is defined as a Bundle ID search string. Each part of an App ID has different and important uses for your app. [Learn More](#)

App ID Description

Name:

You cannot use special characters such as @, #, %, ^, &, _.

Certificates, Identifiers & Profiles

iOS Apps

Certificates

- All
- Pending
- Development
- Production

Identifiers

- App IDs
- Pass Type IDs
- Website Push IDs
- iCloud Containers
- App Groups

Devices

- All
- Apple TV
- Apple Watch
- iPad
- iPhone
- iPod Touch

Provisioning Profiles

- All

Register iOS App ID



ID

Registering an App ID

The App ID string contains two parts separated by a period (.)—an App ID Prefix that is defined as your Team ID by default and an App ID Suffix that is defined as a Bundle ID search string. Each part of an App ID has different and important uses for your app. [Learn More](#)

App ID Description

Name:

 You cannot use special characters such as @, &, *, "

App ID Prefix

Value:

 Z43V935UG2 (Team ID)

App ID Suffix

 Explicit App ID

If you plan to incorporate app services such as Game Center, In-App Purchase, Data Protection, and iCloud, or want a provisioning profile unique to a single app, you must register an explicit App ID for your app.

To create an explicit App ID, enter a unique string in the Bundle ID field. This string should match the Bundle ID of your app.

Bundle ID: com.menard.PushChat

com.menard.PushChat

App Services

Select the services you would like to enable in your app. You can edit your choices after this App ID has been registered.

- Enable Services:
- App Groups
 - Associated Domains
 - Data Protection
 - Complete Protection
 - Protected Unless Open
 - Protected Until First User Authentication
 - Game Center
 - HealthKit
 - HomeKit
 - Wireless Accessory Configuration
 - iCloud
 - Compatible with Xcode 5
 - Include CloudKit support (requires Xcode 6)
 - In-App Purchase
 - Inter-App Audio
 - Wallet
 - Push Notifications
 - VPN Configuration & Control

Cancel

Continue

App ID Description: **PushChat**

Identifier: **Z43V935UG2.com.menard.PushCat**

App Groups: Disabled

Associated Domains: Disabled

Data Protection: Disabled

Game Center: Enabled

HealthKit: Disabled

HomeKit: Disabled

Wireless Accessory Configuration: Disabled

iCloud: Disabled

In-App Purchase: Enabled

Inter-App Audio: Disabled

Wallet: Disabled

Push Notifications: Configurable

VPN Configuration & Control: Enabled

iOS Apps

Certificates

- All
- Pending
- Development
- Production

Identifiers

- App IDs
- Pass Type IDs
- Website Push IDs
- iCloud Containers
- App Groups

Devices

- All
- Apple TV
- Apple Watch
- iPad
- iPhone
- iPod Touch

Provisioning Profiles

- All

iOS App IDs

27 App IDs Total

Name	ID
PushChat	com.menard.PushChat

ID

Name: PushChat
Prefix: Z43V935UG2
ID: com.menard.PushChat

Application Services:

- Service
- App Group

Associated Domains

Data Protection

Game Center

HealthKit

HomeKit

Wireless Accessory Configuration

iCloud

In-App Purchase

Inter-App Audio

Wallet

Push Notifications

VPN Configuration & Control

Push Notifications

Enabled

Push Notifications

Enabled

Push Notifications

Disabled

Edit

+ 🔎

Push Notifications

Enabled

Apple Push Notification service SSL Certificates

To configure push notifications for this iOS App ID, a Client SSL Certificate that allows your notification server to connect to the Apple Push Notification Service is required. Each iOS App ID requires its own Client SSL Certificate. Manage and generate your certificates below.

Development SSL Certificate

Name: Apple Development iOS Push Services: com.menar...
Type: APNs Development iOS
Expires: 30 déc. 2016

Revoke Download

Create an additional certificate to use for this App ID.

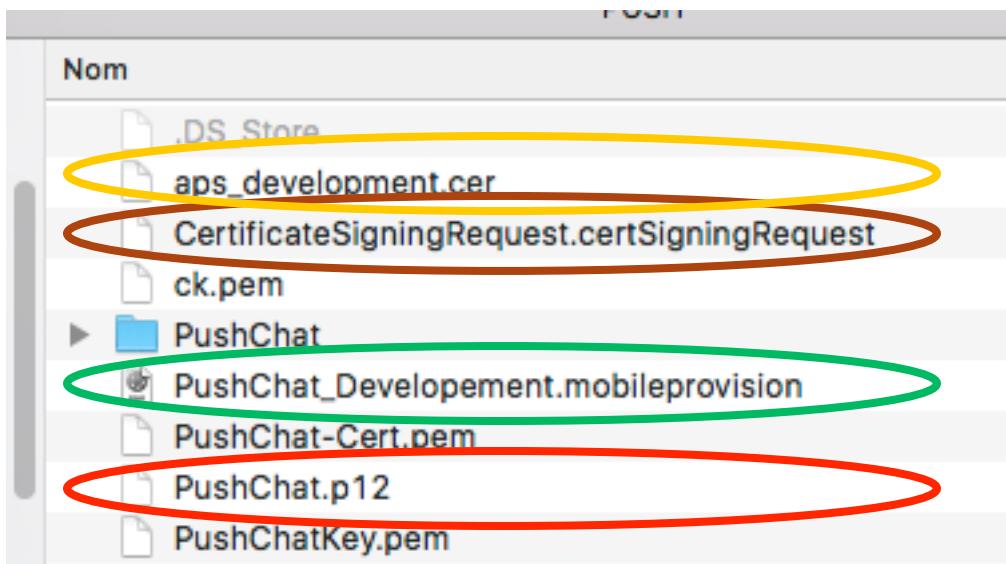
VPN Configuration & Control

Disabled

Delete Done

certificateSigningRequest.certSigningRequest

aps_development.cer



Certificat

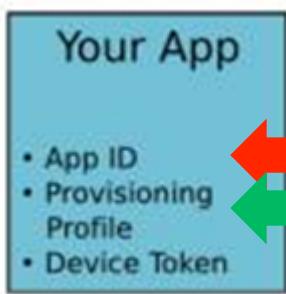
Requête de certificat (CSR / SSL)

Clé privée et clé publique

Profil de provisionnement

Clé publique, format PEM

Clé publique et privée, format p12



Création d'un profil de provisionnement à télécharger dans Xcode et sur le device

Developer Technologies Resources Programs Support Member Center Search Developer

Certificates, Identifiers & Profiles Michel Ménard ▾

iOS Apps

Select Type Configure Generate Download

What type of provisioning profile do you need?



Development

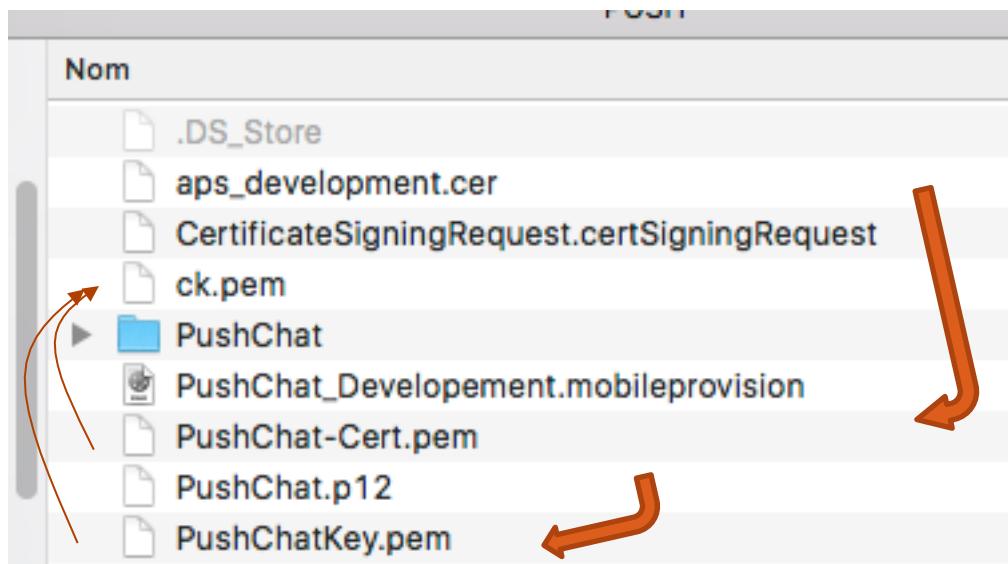
iOS App Development
Create a provisioning profile to install development apps on test devices.

Cancel **Continue**

Provisioning Profiles

All

The 'Provisioning Profiles' section is highlighted with a green oval.



Certificat

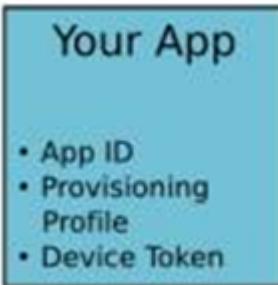
Requête de certificat (CSR / SSL)

Clé privée et clé publique

Profil de provisionnement

Clé publique, format PEM

Clé publique et privée, format p12



Clé privée et certificat pour le serveur (ex: php) : ck.pem

```
openssl x509 -in aps_development.cer -inform der -out PushChatCert.pem
```

```
openssl pkcs12 -nocerts -out PushChatKey.pem -in PushChatKey.p12
```

```
Enter Import Password:
```

```
MAC verified OK
```

```
Enter PEM pass phrase:
```

```
Verifying - Enter PEM pass phrase:
```

```
cat PushChatCert.pem PushChatKey.pem > ck.pem
```

Test

```
openssl s_client -connect gateway.sandbox.push.apple.com:2195  
-cert PushChatCert.pem -key PushChatKey.pem
```

```
Enter pass phrase for PushChatKey.pem:
```

Exemple de token

```
<740f4707 bebcd74f 9b7c25d4 8e335894 5f6aa01d a5ddb387 462c7eaf 61bb78ad>
```

```
teaming:SimplePush mmenard$ openssl s_client -connect gateway.sandbox.push.apple.com:2195 -cert PushChatCert.pem -key|
```

```
PushChatKey.pem
```

```
Enter pass phrase for PushChatKey.pem:
```

```
CONNECTED(00000003)
```

```
depth=2 O = Entrust.net, OU = www.entrust.net/CPS_2048 incorp. by ref. (limits liab.), OU = (c) 1999 Entrust.net Limited, CN = Entrust.net Certification Authority (2048)
```

```
verify return:1
```

```
depth=1 C = US, O = "Entrust, Inc.", OU = See www.entrust.net/legal-terms, OU = "(c) 2012 Entrust, Inc. - for authorized use only", CN = Entrust Certification Authority - L1K
```

```
verify return:1
```

```
depth=0 C = US, ST = California, L = Cupertino, O = Apple Inc., CN = gateway.sandbox.push.apple.com
```

```
verify return:1
```

```
---
```

Certificate chain

```
0 s:/C=US/ST=California/L=Cupertino/O=Apple Inc./CN=gateway.sandbox.push.apple.com
```

```
  i:/C=US/O=Entrust, Inc./OU=See www.entrust.net/legal-terms/OU=(c) 2012 Entrust, Inc. - for authorized use only/CN=Entrust Certification Authority - L1K
```

```
1 s:/C=US/O=Entrust, Inc./OU=See www.entrust.net/legal-terms/OU=(c) 2012 Entrust, Inc. - for authorized use only/CN=Entrust Certification Authority - L1K
```

```
  i:/O=Entrust.net/OU=www.entrust.net/CPS_2048 incorp. by ref. (limits liab.)/OU=(c) 1999 Entrust.net Limited, CN = Entrust.net Certification Authority (2048)
```

```
---
```

Server certificate

```
-----BEGIN CERTIFICATE-----
```

```
MIIxFUTCCBDMgAwIBAgIRAP/KN+WwyNu6AAAAAFDYCGYwDQYJKoZIhvcNAQELBQA  
gboxCzAJBgNVBAYTAIVTMRYwFAYDVQQKEw1FbnRydXN0LCBjbmMuMSgwJgYDVQQ  
Ex9TZWUgd3d3LmVudHJ1c3QubmV0L2xlZ2FsLXRlcmtzMTkwNwYDVQQLEzAoYykg  
MjAxMiBFbnRydXN0LCBjbmMuIC0gZm9yIGF1dGhvcml6ZWQgdXNlIG9ubHkxLjAs
```

```
Server public key is 2048 bit
```

```
Secure Renegotiation IS supported
```

```
Compression: NONE
```

```
Expansion: NONE
```

```
No ALPN negotiated
```

```
SSL-Session:
```

```
  Protocol : TLSv1.2
```

```
  Cipher   : AES256-SHA
```

```
  Session-ID:
```

```
  Session-ID-ctx:
```

```
  Master-Key:
```

```
  C5C07FCABC3AE7488F69F2216C52EDA5DB20
```

```
  Key-Ag : None
```

```
  PSK identity: None
```

```
  PSK identity hint: None
```

```
  SRP username: None
```

```
  Start Time: 1483009875
```

```
  Timeout : 300 (sec)
```

```
  Verify return code: 0 (ok)
```

```
---
```

The screenshot shows the Xcode interface with the following details:

- Title Bar:** PushChat (iPhone4S1)
- Toolbar:** Standard Xcode toolbar.
- Project Navigator:** Shows the project structure under "PushChat".
- File Navigator:** Shows the file "AppDelegate.m" is selected.
- Editor:** Displays the code for `AppDelegate.m`. The code handles remote notifications and logs the device token.

```
// AppDelegate.m
// PushChat
//
// Created by Michel Ménard on 31/12/2015.
// Copyright © 2015 Michel Ménard. All rights reserved.

#import "AppDelegate.h"

@interface AppDelegate : NSObject<UIApplicationDelegate>

@end

@implementation AppDelegate

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after application launch.
    [[UIApplication sharedApplication] registerForRemoteNotificationTypes:(UIRemoteNotificationTypeBadge | UIRemoteNotificationTypeSound | UIRemoteNotificationTypeAlert)];

    return YES;
}

- (void) application:(UIApplication *)application didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken
{
    NSLog(@"My token is : %@",deviceToken);
}

-(void) application:(UIApplication *)application didFailToRegisterForRemoteNotificationsWithError:(NSError *)error
{
    NSLog(@"Failed to get token, error:%@",error);
}


```

2016-01-05 17:01:16.158 PushChat[554:60b] My token is :
<da39b36a 2f05a2b0 bc12a055 c84778f6

IOS 8

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after application launch.

    // [[UIApplication sharedApplication] registerForRemoteNotificationTypes:(UIRemoteNotificationTypeBadge | UIRemoteNotificationTypeSound | UIRemoteNotificationTypeAlert)];

    if ([[UIDevice currentDevice] systemVersion] floatValue] >= 8.0)
    {
        [[UIApplication sharedApplication] registerUserNotificationSettings:[UIUserNotificationSettings settingsForTypes:(UIUserNotificationTypeSound | UIUserNotificationTypeAlert | UIUserNotificationTypeBadge) categories:nil]];
        [[UIApplication sharedApplication] registerForRemoteNotifications];
    }
    else
    {
        [[UIApplication sharedApplication] registerForRemoteNotificationTypes:
         (UIUserNotificationTypeBadge | UIUserNotificationTypeSound | UIUserNotificationTypeAlert)];
    }

    return YES;
}
```

```
solr:SimplePush michelmenard$ php simplepush.php
Connected to APNS
Message successfully delivered
solr:SimplePush michelmenard$ more simplepush.php
<?php

// Put your device token here (without spaces):
$deviceToken = 'da39b36a2f05a2b0bc12a055c84'
// Put your private key's passphrase here:
$passphrase = 'pushchat';

// Put your alert message here:
$message = 'My first push notification!';

///////////////////////////////[Restauré]
$ctx = stream_context_create();
stream_context_set_option($ctx, 'ssl', 'local_cert', 'ck.pem');
stream_context_set_option($ctx, 'ssl', 'passphrase', $passphrase);
// Open a connection to the APNS server
$fp = stream_socket_client(
    'ssl://gateway.sandbox.push.apple.com:2195', $err,
    $errstr, 60, STREAM_CLIENT_CONNECT|STREAM_CLIENT_PERSISTENT, $ctx);
if (!$fp)
    exit("Failed to connect: $err $errstr" . PHP_EOL);
echo 'Connected to APNS' . PHP_EOL;

// Create the payload body
$body['aps'] = array(
    'alert' => $message,
    'sound' => 'default'
);

// Encode the payload as JSON
	payload = json_encode($body);
[

// Build the binary notification
$msg = chr(0) . pack('n', 32) . pack('H*', $deviceToken) . pack('n', strlen($payload)) . $payload;
[

// Send it to the server
$result = fwrite($fp, $msg, strlen($msg));
if (!$result)
    echo 'Message not delivered' . PHP_EOL;
else
    echo 'Message successfully delivered' . PHP_EOL;

// Close the connection to the server
fclose($fp);[Restauré]
mapLP-Info.plist
mapLP-Prefix.pch
SecondViewController.h
SecondViewController.m
toto.png
project.pbxproj
contents.xcworkspace.xcuserdata
UserInterfaceState.xcuserstate
UserInterfaceState.xcuserstate
Breakpoints_v2.xcbkptlist
mapLP.xcscheme
xcschememanagement.plist
mapLP.xcscheme
mapLP.xcscheme
xcschememanagement.plist
mapLPTests.m
gnosia:~ ochappe$ Write failed: Broken pipe
solr:~ michelmenard$[Restauré 3 déc. 2015 17:51:21]
solr:ships michelmenard$ Last login: Thu Dec 3 17:51:17 on ttys005
[Restauré 31 déc. 2015 11:05:45]
solr:~ michelmenard$ Last login: Tue Dec 29 09:05:22 on console
Restored session: Ven 4 déc 2015 14:52:35 CET-01:00
solr:~ michelmenard$ [Restauré 5 janv. 2016 17:01:28]
solr:ships michelmenard$ Last login: Sat Jan 2 11:05:42 on console
Restored session: Jeu 31 déc 2015 16:25:25 CET-01:00
solr:~ michelmenard$ [Restauré]
solr:ships michelmenard$ Last login: Mon Nov 23 21:46:34 on ttys027
[Restauré]
solr:ships michelmenard$ Last login: Wed Dec 2 23:10:31 on console
[Restauré]
solr:ships michelmenard$ Last login: Thu Dec 3 17:51:31 on ttys021
[Restauré 3 déc. 2015 17:51:30]
solr:ships michelmenard$ Last login: Thu Dec 3 17:51:31 on ttys021
[Restauré]
solr:ships michelmenard$ Restored session: Ven 4 déc 2015 14:52:35 CET-01:00
solr:~ michelmenard$ [Restauré 5 janv. 2016 11:05:46]
solr:ships michelmenard$ Last login: Tue Dec 29 09:05:22 on console
Restored session: Ven 4 déc 2015 14:52:35 CET-01:00
solr:~ michelmenard$ [Restauré]
solr:ships michelmenard$ Last login: Sat Jan 2 11:05:42 on console
Restored session: Jeu 31 déc 2015 16:25:25 CET-01:00
```

*Ce que vous devez
(s)avoir avant de
commencer ...*



*Premiers projets:
Hello
Application
interactive*

Gestion des vues

*Plus loin...
Autorotation
Photo
Audio, vidéo*

III Complément sur objective C

Protocole et gestion mémoire



Protocoles

L'héritage multiple étant impossible en Objective-C, un mécanisme est ajouté permettant de définir des protocoles qu'une classe s'engage à respecter.

```
@protocol MonProtocole  
- (void)uneMethodeDuProtocole  
  
@optional  
// Déclaration des méthodes optionnelles  
@end
```

Définition d'un protocole dans un fichier d'entête : ressemble à la déclaration d'une classe mais sans variable d'instance.

```
@interface MatroisiemeClasse: NSObject <MonProtocole, MonAutreProtocole>{  
// Variables d'instance  
}  
  
@end
```

Une classe déclare dans son interface la liste des protocoles qu'elle implémente.

Classe GestionNotes et protocole GestionNotesDelegate .h

```
#import <Foundation/Foundation.h>

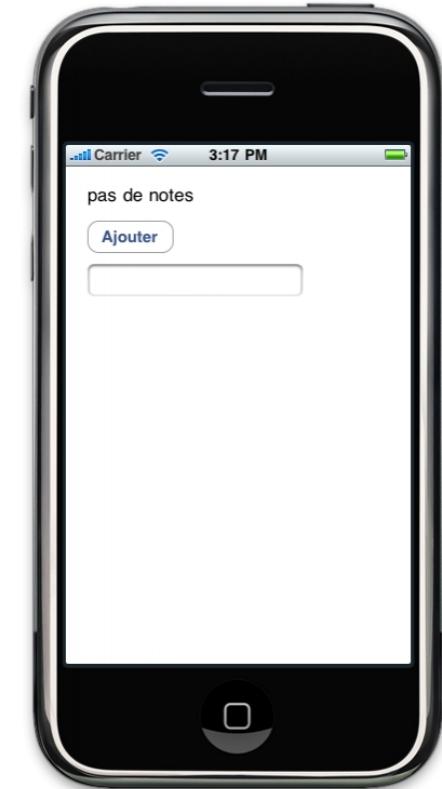
@protocol GestionNotesDelegate;

@interface GestionNotes : NSObject {
    id<GestionNotesDelegate> delegate; // 1
    NSString *studentName;
    NSMutableArray *arrayDeNotes;
}

@property (nonatomic, weak) id<GestionNotesDelegate> delegate; // 2
@property (nonatomic, copy) NSString *studentName;
- (id) initWithStudentName:(NSString*)name;
- (void) ajouterNoteALaMoyenne:(NSInteger)note;
@end

@protocol GestionNotesDelegate // 3
@required // 4
-(void) gestionNotes:(GestionNotes*)gestionNotes aReceuUneNote:(NSInteger)note
                  nouvelleMoyenne:(float)moyenne; // 5

@optional // 6
- (void) gestionNotes:(GestionNotes*)gestionNotes enDessousMoyenne:(float)moyenne; // 7
@end
```



```

#import "GestionNotes.h"

@interface GestionNotes (PrivateMethods)
- (float) calculMoyenne; // 2
@end

@implementation GestionNotes
@synthesize delegate, studentName;

- (id) initWithStudentName:(NSString *)name {
self = [super init]; // 3
if (self)
{
self.studentName = name; // 4
arrayDeNotes = [[NSMutableArray alloc] init];
}
return self;
}

- (void) ajouterNoteALaMoyenne:(NSInteger)note {
NSNumber *noteNumber = [NSNumber numberWithInteger:note]; // 5
[arrayDeNotes addObject:noteNumber];
float moyenne = [self calculMoyenne]; // 6

if (moyenne < 10.0 && [delegate respondsToSelector:@selector(gestionNotes:enDessousMoyenne:)]) // 7
{
[delegate gestionNotes:self enDessousMoyenne:moyenne]; // 8
}
[delegate gestionNotes:self aRecuUneNote:note nouvelleMoyenne:moyenne]; // 9
}
@end

@implementation GestionNotes (PrivateMethods) // 10
- (float) calculMoyenne {
NSInteger cumulNotes = 0;

for (NSNumber *number in arrayDeNotes) {
NSInteger integerValue = [number integerValue]; // 11
cumulNotes += integerValue; // 12
}
return (float)cumulNotes/(float)[arrayDeNotes count]; // 13
}
@end

```

Classe GestionNotes et protocole GestionNotes Delegate .m



```

- (void) dealloc {
[arrayDeNotes release];
[studentName release];
[super dealloc];
}

```

Classe ViewController : .h (le délégué)

||| Exemple Protocole ...

```
#import <UIKit/UIKit.h>
#import "GestionNotes.h" // 1

@interface RootViewController : UIViewController <GestionNotesDelegate> { // 2

IBOutlet UILabel *labelNote;
IBOutlet UIButton *buttonValider;
IBOutlet UITextField *textFieldNote;

GestionNotes *gestionnaire; // 3

}

- (IBAction) validerSaisieNote:(id)sender;

@end
```



Classe ViewController : .m (le délégué)

```
- (void)viewDidLoad
{
[super viewDidLoad];
textFieldNote.keyboardType = UIKeyboardTypeNumberPad;
textFieldNote.clearsOnBeginEditing = YES;

gestionnaire = [[GestionNotes alloc]
initWithStudentName:@"Titi"]; // 4
gestionnaire.delegate = self; // 5
}

#pragma mark -
#pragma Actions

- (IBAction) validerSaisieNote:(id)sender {
// on rentre le clavier
NSInteger note = [textFieldNote.text integerValue];

if (note <= 20) { // 6
[gestionnaire ajouterNoteALaMoyenne:note];
[textFieldNote resignFirstResponder];
}
else
{
textFieldNote.text = @""; // 7
}
}
```



Classe ViewController : .m

```
#pragma mark - // 8
#pragma mark GestionNotesDelegate methods

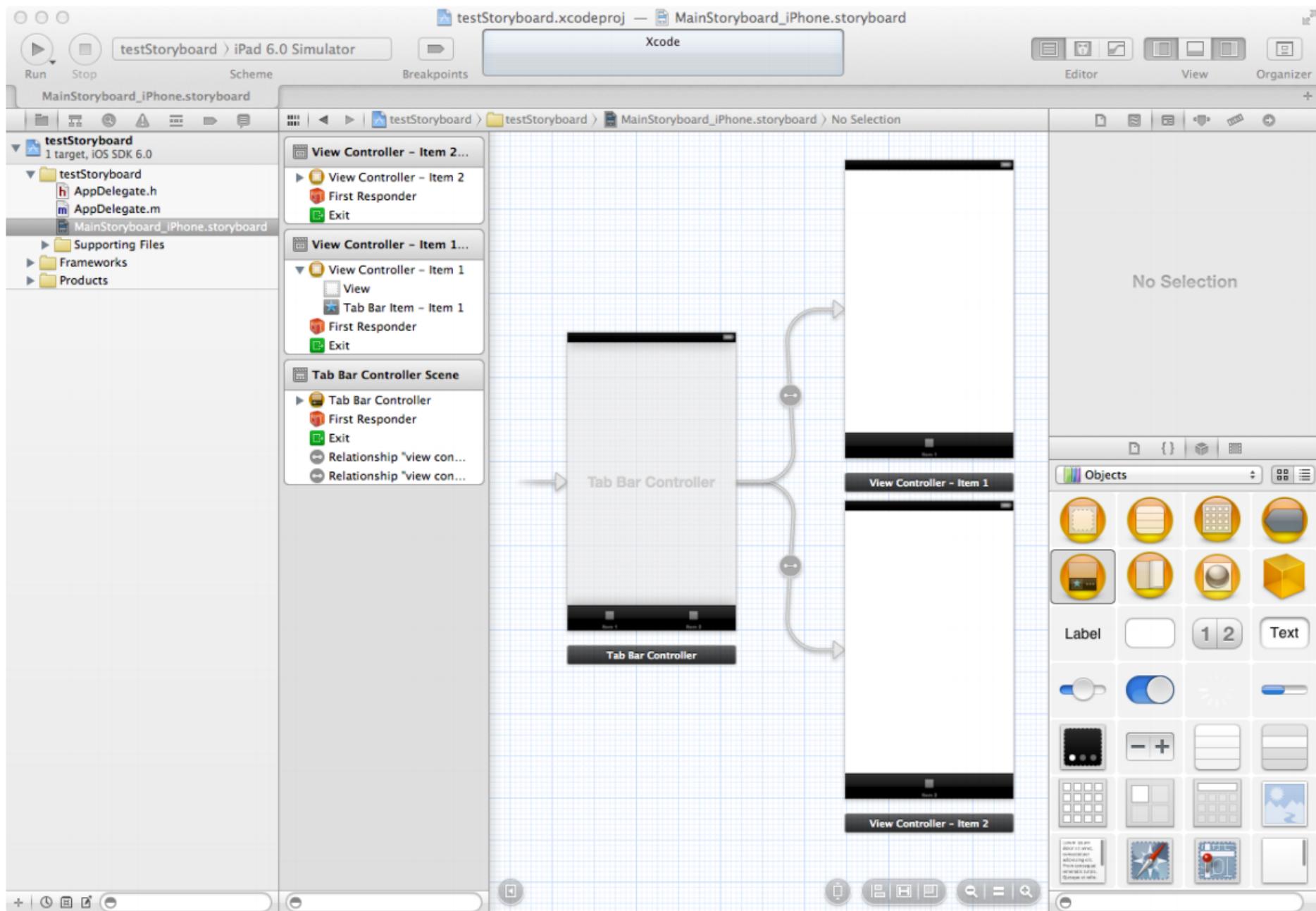
- (void) gestionNotes:(GestionNotes *)gestionNotes aReçuUneNote:
(NSInteger)note nouvelleMoyenne:(float)moyenne {
[labelNote setText:[NSString stringWithFormat:@"Moyenne : %.2f",
moyenne]]; // 9
}

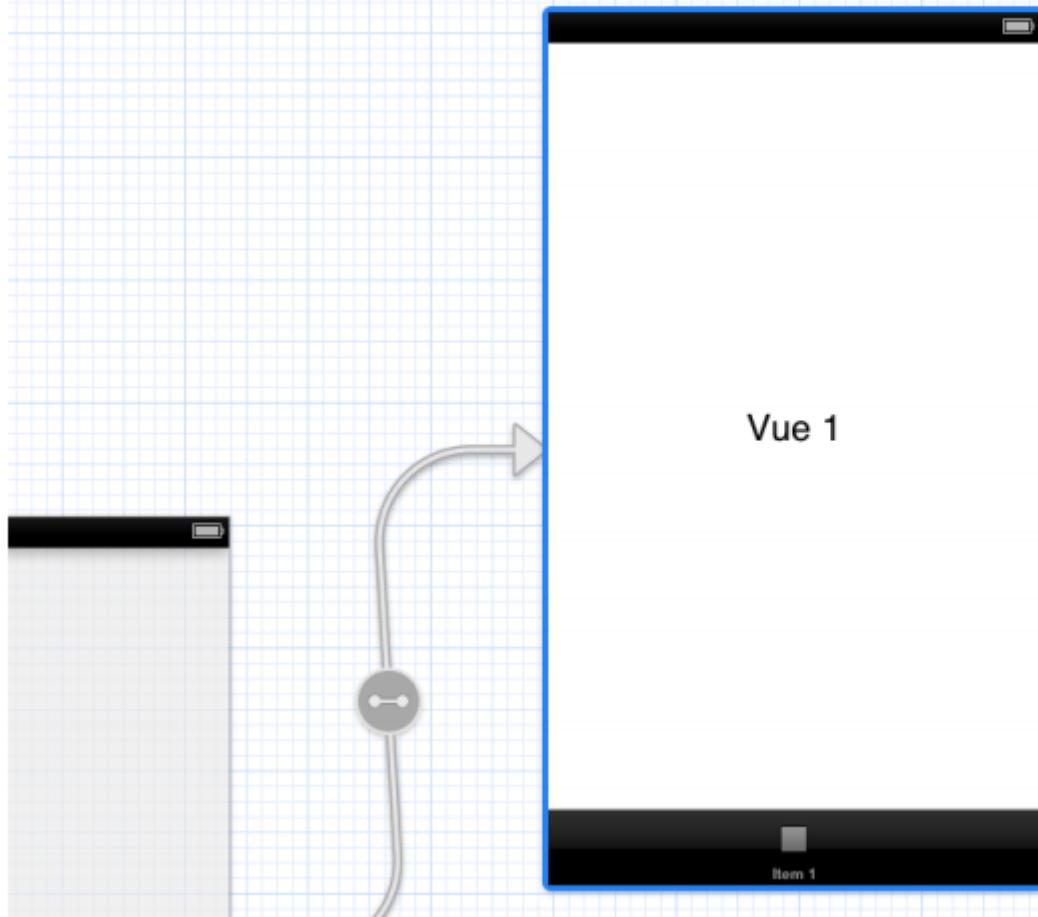
- (void) gestionNotes:(GestionNotes *)gestionNotes enDessousMoyenne:
(float)moyenne {
UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"Attention !"
message:[NSString stringWithFormat:@"Votre moyenne de %f n'est pas
bonne !", moyenne]
delegate:self
cancelButtonTitle:@"Ok"
otherButtonTitles:nil]; // 10
[alert show];
[alert release];
}
```

N'oubliez pas le dealloc !

```
- (void) dealloc
{
[gestionnaire release];
[super dealloc];
}
```





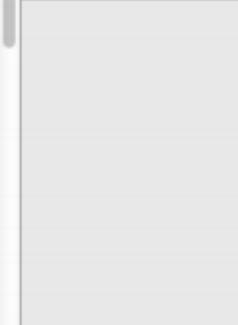


Custom Class
Class item1ViewController

Identity
Storyboard ID
Restoration ID Use Storyboard ID

User Defined Runtime Attributes
Key Path Type Value

+ - Document
Label Xcode Specific Label
Object ID GZy-F0-Eju
Lock Inherited - (Nothing)
Notes
No Font



*Ce que vous devez
(s)avoir avant de
commencer ...*

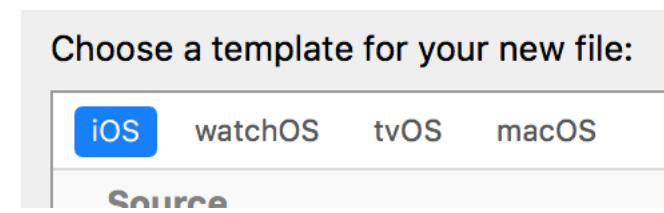
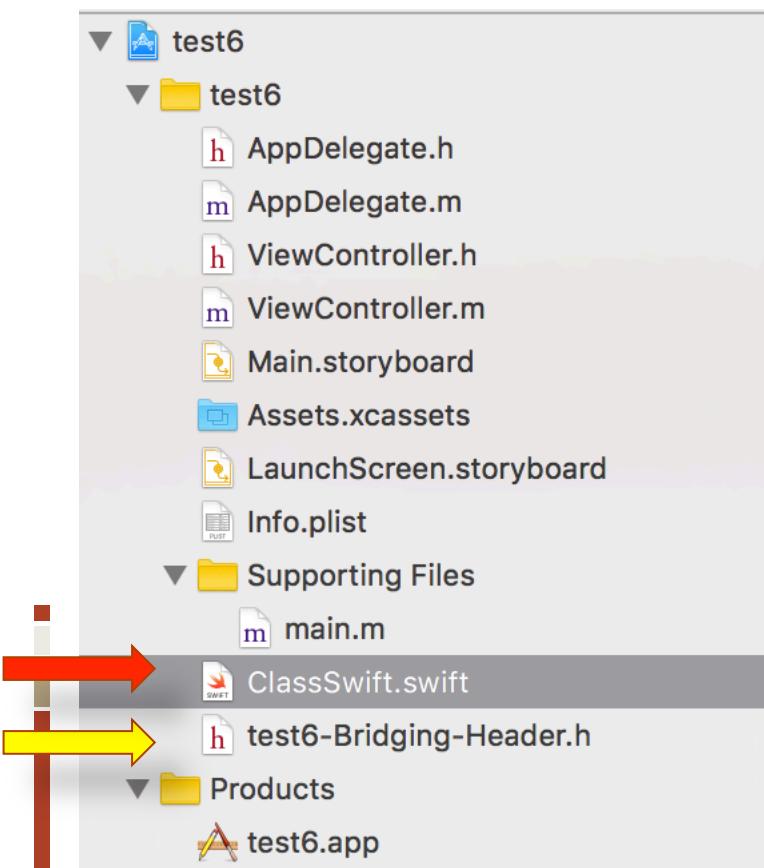


*Premiers projets:
Hello
Application
interactive*

*Quelques mots
Swift*

*Plus loin...
Autorotation
Photo
Audio, vidéo*

Une classe swift dans un projet objective-C

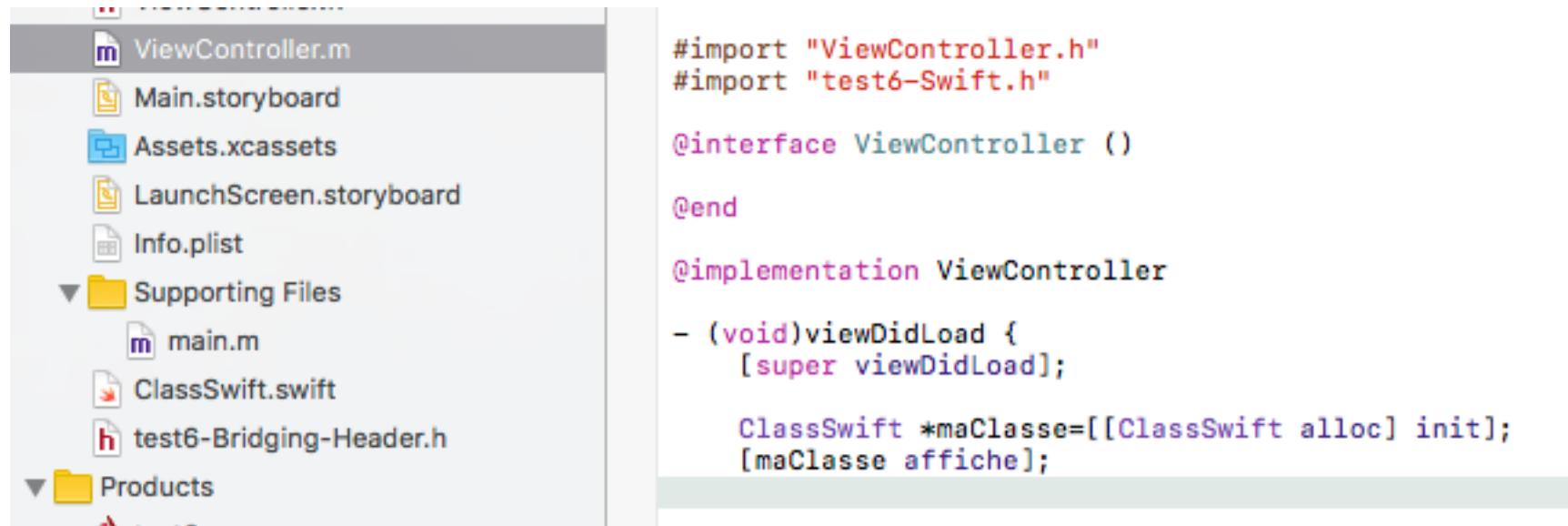


```
import Foundation

@objc class ClassSwift: NSObject{
    func affiche(){
        print("toto")
    }
}
```

```
//
// Use this file to im
//
```

||| Une classe swift dans un projet objective-C



The image shows a screenshot of the Xcode interface. On the left, the Project Navigator displays the project structure:

- ViewController.m
- Main.storyboard
- Assets.xcassets
- LaunchScreen.storyboard
- Info.plist
- Supporting Files
 - main.m
 - ClassSwift.swift
 - test6-Bridging-Header.h
- Products

On the right, the code editor shows the implementation of `ViewController.m`:

```
#import "ViewController.h"
#import "test6-Swift.h"

@interface ViewController : UIViewController

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];

    ClassSwift *maClasse=[[ClassSwift alloc] init];
    [maClasse affiche];
}


```

Une classe swift dans un projet objective-C

The screenshot shows the Xcode interface with a project named "test6". The left sidebar displays the project structure:

- test6 (target)
- test6 (group)
 - AppDelegate.h
 - AppDelegate.m
 - ViewController.h
 - ViewController.m
 - Main.storyboard
 - Assets.xcassets
 - LaunchScreen.storyboard
 - Info.plist
- Supporting Files
 - main.m
 - ClassSwift.swift
 - test6-Bridging-Header.h
- Products
 - test6.app

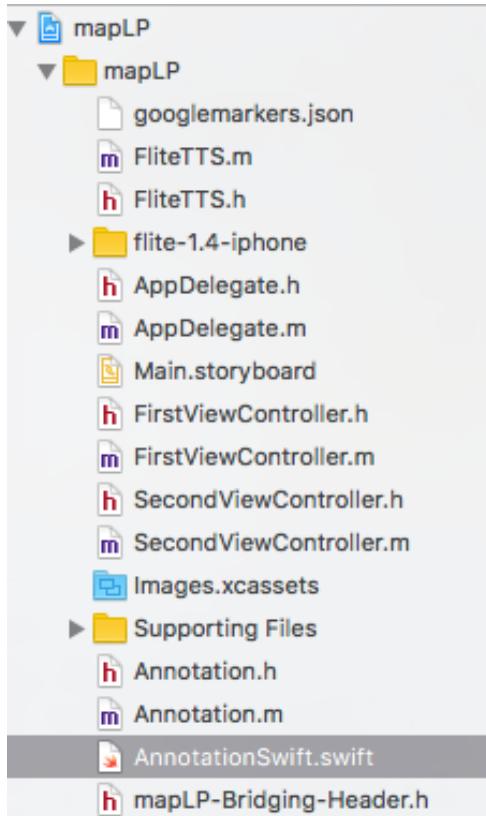
The main area shows the "Build Settings" tab for the "test6" target. The "Swift Compiler - General" section is highlighted with a red box and contains the following settings:

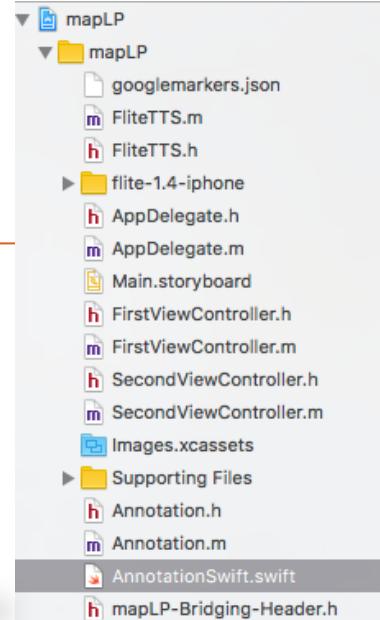
Install Objective-C Compatibility Header	Yes
Objective-C Bridging Header	test6/test6-Bridging-Header.h
Objective-C Generated Interface Header Name	test6-Swift.h
Reflection Metadata Level	All



toto

Une classe swift dans un projet objective-C (2^{ème} exemple:TP)





```
import Foundation
import MapKit
```

```
@objc class AnnotationSwift: NSObject, MKAnnotation {
```

```
    let title: String?
    let locationName: String
    let discipline: String
    let coordinate: CLLocationCoordinate2D
```

```
    init(title: String, locationName: String, discipline: String, coordinate: CLLocationCoordinate2D) {
```

```
        self.title = title
        self.locationName = locationName
        self.discipline = discipline
        self.coordinate = coordinate
```

```
        super.init()
    }
```

```
    var subtitle: String? {
        return locationName
    }
}
```

```
import Foundation
import MapKit
// MKAnnotation.h
// MapKit
// Copyright (c) 2009-2014, Apple Inc. All rights reserved.
//public protocol MKAnnotation : NSObjectProtocol {

    // Center latitude and longitude of the annotation view.
    //The implementation of this property must be KVO compliant.
    public var coordinate: CLLocationCoordinate2D { get }

    //Title and subtitle for use by selection UI.

    optional public var title: String? { get }
    optional public var subtitle: String? { get }
}
```

Une classe swift dans un projet objective-C

```
import Foundation
import MapKit

@objc class AnnotationSwift: NSObject, MKAnnotation {
    let title: String?
    let locationName: String
    let discipline: String
    let coordinate: CLLocationCoordinate2D

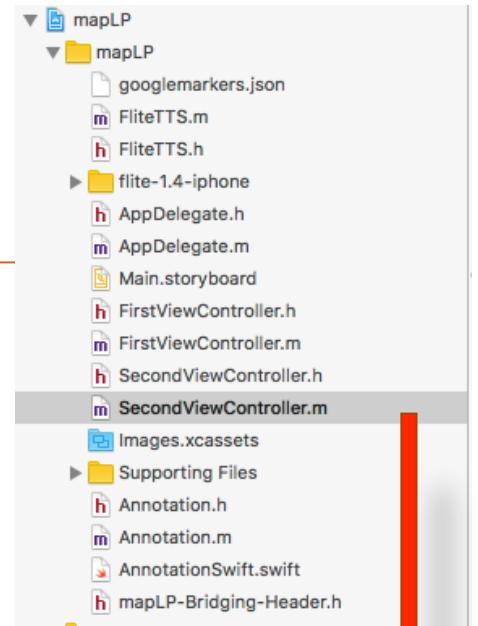
    init(title: String, locationName: String, discipline: String, coordinate: CLLocationCoordinate2D) {
        self.title = title
        self.locationName = locationName
        self.discipline = discipline
        self.coordinate = coordinate

        super.init()
    }

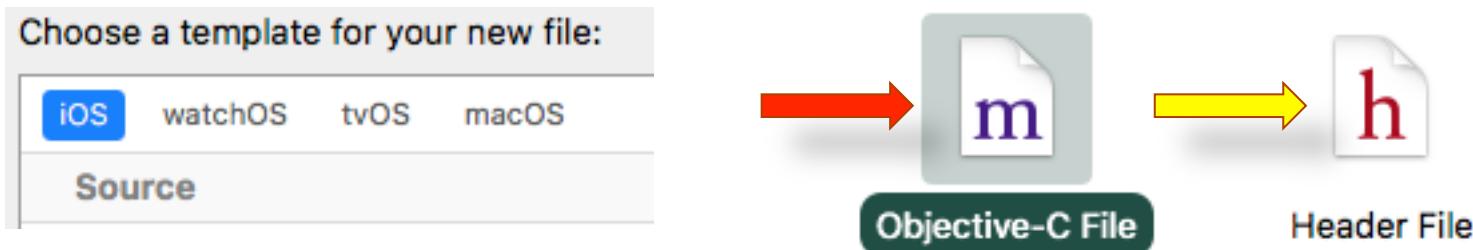
    var subtitle: String? {
        return locationName
    }
}
```

```
#import "mapLP-swift.h"
```

```
AnnotationSwift *annotationSwift=[AnnotationSwift alloc];
annotationSwift=[annotationSwift initWithTitle:@"Port
des minimes" locationName: @"Capitainerie" discipline:
@"Plaisance" coordinate:region.center];
[self.myMapView addAnnotation:annotationSwift];
```



Une classe objective-C dans un projet swift



The screenshot shows the Xcode interface with a project named 'test4'. The left sidebar shows the file structure:

- test4
- └ test4
 - Book.swift
 - AppDelegate.swift
 - MasterViewController.swift
 - DetailViewController.swift
 - Main.storyboard
 - Assets.xcassets
 - LaunchScreen.storyboard
 - Info.plist
 - myClass.m
 - test4-Bridging-Header.h
 - myClass.h
- Products

A red arrow points to the 'myClass.m' file in the sidebar. The main editor area displays the contents of 'myClass.m':

```
// myClass.m
// test4
//
// Created by Michel Menard on 21/10/2016.
// Copyright © 2016 Michel Menard. All rights reserved.

#import "myClass.h"
@implementation myClass

- (void) sayIt{
    NSLog(@"Bonjour");
}

@end
```

A yellow arrow points to the 'myClass.h' file in the sidebar. The right editor area displays the contents of 'myClass.h':

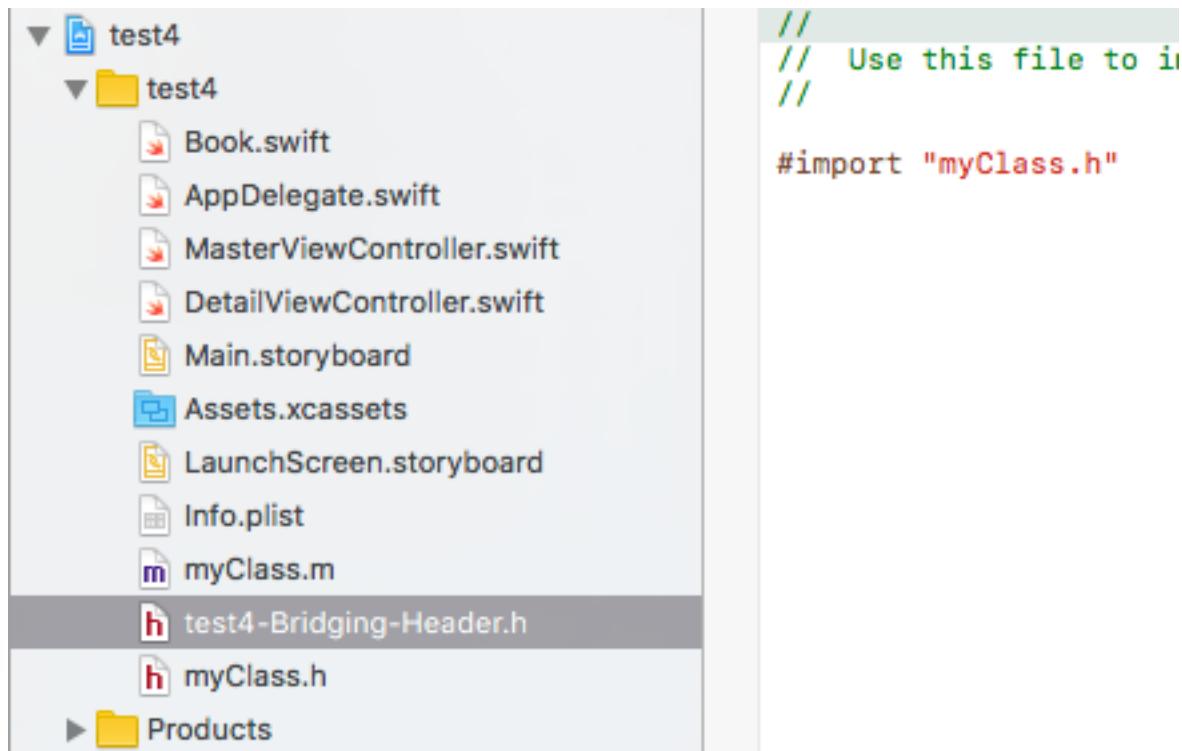
```
#import <Foundation/Foundation.h>

@interface myClass : NSObject

- (void) sayIt;

@end
```

||| Une classe objective-C dans un projet swift



||| Une classe objective-C dans un projet swift



```
override func viewDidLoad() {
    super.viewDidLoad()
    // Do any additional setup after loading the view,
    self.navigationItem.leftBarButtonItem = self.editButtonItem

    let addButton = UIBarButtonItem(barButtonSystemItem: UIBarButtonSystemItem.insertNewObject(_:))
    self.navigationItem.rightBarButtonItem = addButton
    if let split = self.splitViewController {
        let controllers = split.viewControllers
        self.detailViewController = (controllers[controllers.count - 1] as? DetailViewController)
    }
    let maclasse=myClass()
    maclasse.sayIt()
}
```

Une classe objective-C dans un projet swift

The screenshot shows the Xcode interface with a Swift project named "test4". The "myClass.m" file is selected in the left sidebar, and its contents are displayed in the main editor area.

```
// myClass.m
// test4
//
// Created by Michel Menard on 21/10/2016.
// Copyright © 2016 Michel Menard. All rights reserved.

#import "myClass.h"
@implementation myClass

- (void) sayIt{
    NSLog(@"Bonjour");
}

@end
```

Swift : variable et constante

var name : *Type* = *value*

Variable

```
var valueThatMayChange = "Hello "
```

let name : *Type* = *value*

Constant

```
let valueThatWillNotChange = "Hello world"
```

var name : *Type?* = *value*

```
var anEmptyStringForNow: String?  
var myConditionalInt: Int?  
myConditionalInt = 1  
myConditionalInt = nil
```

var name : *Type!* = *value*

```
var myInt1: Int? = 1  
var myInt2: Int? = 2  
let sum = myInt1! + myInt2!
```

let name : *Type!* = *value*

```
var myInt1: Int!  
myInt1 = 1  
let myInt2: Int! = 2  
let sum = myInt1 + myInt2
```

Swift : variable et constante - optionnelles

```
import UIKit

var str = "Hello, playground"

var toto:Int?
var tata:Int?
var tutu:Int?

//tutu=toto+tata
if let test=toto {
    if let test1=tata {
        tutu=test+test1
    }
}
print("==> \(tutu)")
```

"Hello, playground"

nil
nil
nil

"==> nil\n"

```
import UIKit

var str = "Hello, playground"

var toto:Int?
var tutu:Int?
var titi:Int?

tutu=toto!+titi!
```

let const:Int?
const=3
let const1=const!+const!

3
6

let const2:Int!
const2=3
let const3=const2+const2

3
6

! error: Execution was interrupted, reason: EXC_BAD_INSTRUCTION (code=EXC_I386_INVOP, subcode=0x0).

Swift : variable et constante – optionnelles

```
struct Rectangle{  
    var width:Int=0  
    var height:Int=0  
  
    /* Propriété calculée */  
    var aera:Int{  
        return self.width*self.height  
    }  
  
    /* Méthode d'instance */  
    func draw(){  
        print("Drawing a \(self.width) by \(self.height)")  
    }  
  
    /* Méthode statique */  
    static func buildRegularSquare()->Rectangle{  
        return Rectangle(width:100,height:100)  
    }  
}
```

```
/* Structure */  
let rect=Rectangle(width:640,height:480)  
let rect1=Rectangle()  
rect1.aera  
  
var carre=Rectangle.buildRegularSquare()
```

Swift : variable et constante - optionnelles

```
struct RectangleX{
    var width:Int?
    var height:Int?

    var area:Int? {
        if let lwidth=self.width {
            if let lheight=self.height{
                return lwidth*lheight
            }
            else {return nil}
        }
        else {return nil}
    }

    /* Méthode d'instance */
    func draw(){
        print("Drawing a \(\self.width) by \(\self.height)")
    }
}
```

Comment simplifier ?

Swift : variable et constante - optionnelles

```
struct RectangleX{
    var width:Int?
    var height:Int?

    var aera:Int? {
        var laera:Int?
        if let lwidth=width {
            if let lheight=height {
                laera=lwidth*lheight
            }
        }
        return laera
    }

    func draw() {
        print("Drawing a \(self.width) by \(self.height)")
    }
}
```

```
var rect = RectangleX()
rect.aera
var rect1 = RectangleX(width:100,height:100)
rect1.aera
```

```
RectangleX
nil
RectangleX
10000
```

Swift : variable et constante - optionnelles

```
struct RectangleX{  
    var width:Int?  
    var height:Int?  
  
    var aera:Int? {  
        var laera:Int?  
        if let lwidth=width {  
            if let lheight=height {  
                laera=lwidth*lheight  
            }  
        }  
        return laera  
    }  
  
    func draw() {  
        print("Drawing a \(self.width) by \(self.height)")  
    }  
}
```

non optionnelle

```
var rectX:RectangleX?  
let aera=rectX?.aera  
rectX?.draw()  
rectX?.aera  
  
var rect2:RectangleX  
let aera2=rect2.aera  
rect2.draw()  
rect2.aera
```

! Variable 'rect2' used before being initialized

nil
nil
nil
nil

```
var rectXY=RectangleX(width:10,height:10)  
let w=rectXY.width  
rectXY.draw()
```

Swift : variable et constante - optionnelles

```
struct RectangleX{  
    var width:Int?  
    var height:Int?  
  
    var aera:Int? {  
        var laera:Int?  
        if let lwidth=width {  
            if let lheight=height {  
                laera=lwidth*lheight  
            }  
        }  
        return laera  
    }  
  
    func draw() {  
        print("Drawing a \(self.width) by \(self.height)")  
    }  
}
```

```
var rectX:RectangleX?  
let width=rectX?.width  
rectX?.draw()  
rectX?.aera
```

Optionnelle
unwrapping

```
var rectXYZ:RectangleX  
if let imput=rectX {  
    rectXYZ=imput  
}
```

Swift : variable et constante - optionnelles

```
struct RectangleX{  
    var width:Int?  
    var height:Int?  
  
    var aera:Int? {  
        var laera:Int?  
        if let lwidth=width {  
            if let lheight=height {  
                laera=lwidth*lheight  
            }  
        }  
        return laera  
    }  
  
    func draw() {  
        print("Drawing a \(self.width) by \(self.height)")  
    }  
}
```

```
var rectX:RectangleX?  
let width=rectX?.width  
rectX?.draw()  
rectX?.aera
```

Implicity unwrapped

```
var rectXXY:RectangleX?  
rectXXY=RectangleX(width:10,height:10)  
let ww=rectXXY!.width
```

Swift : variable et constante – optionnelles – un dernier exemple

```
class house {  
    var surface:Int?  
    var nbrRoom:Int?  
  
    init(){  
  
        init(surface:Int,nbrRoom:Int){  
            self.surface=surface  
            self.nbrRoom=nbrRoom  
        }  
    }  
}
```

```
var maison=house()  
print("\(maison.surface)")  
if let test=maison.surface  
{  
    var surface=test+100  
}  
maison.surface=100  
maison.nbrRoom=10  
print("\(maison.surface)")  
var chateau=house(surface: 1000,nbrRoom: 500)
```

```
house  
"nil\n"  
  
house  
house  
"Optional(100)\n"  
house
```

Swift : variable et constante – optionnelles – un dernier exemple

```
class house {  
    var surface:Int?  
    var nbrRoom:Int?  
  
    init(){self.surface=0}  
  
    init(surface:Int,nbrRoom:Int){  
        self.surface=surface  
        self.nbrRoom=nbrRoom  
    }  
}
```

```
var maison=house()  
print("\(maison.surface)")  
if let test=maison.surface  
{  
    var surface=test+100  
}  
maison.surface=100  
maison.nbrRoom=10  
print("\(maison.surface)")  
var chateau=house(surface: 1000,nbrRoom: 500)
```

house
"Optional(0)\n"

100 ←

house
house
"Optional(100)\n"
house

Swift :

Classes, propriétés, et fonctions de classe

```
class Person {  
    var name:String="Name"  
    var age:Int=0  
  
    init(){}
  
  
    init(name:String, age:Int){  
        self.name=name  
        self.age=age  
    }
  
  
    func profile()->String{  
        return "I'am \(self.name) and I'm \(self.age) years old"  
    }
}
```

var p = Person()
p.name = "Matt"
p.age = 40
print("p.name = \(p.name)")
print("p.age = \(p.age)")

Propriétés stockées

Deux constructeurs déclarés, dont un sans paramètre

Méthode d'instance

print(p.profile())

I'm Matt and I'm 40 years old.

```
class Person1 {  
  
    init(){}
    var name:String="Name"
    var age:Int=0  
  
    private var _prenom:String="Prénom" {
        didSet{
            print("old value is \(oldValue), new value is \(prenom)")
        }
        willSet{
            print("old value is \(prenom), new value is \(newValue)")
        }
    }
}
```

Variable privée

```
var prenom:String {
    get {
        return _prenom
    }
    set (anewValue) {
        if(anewValue != prenom) {
            _prenom=anewValue
        }
    }
}
```

Getter et Setter :
utilisent la variable locale
On peut les utiliser comme
des propriétés

```
func profile() -> String{
    return "I'am \(self.name) and I'm \(self.age) years old"
}
```

```

class Person1 {

    init(){}
    var name:String="Name"
    var age:Int=0

    private var _prenom:String="Prénom" {
        didSet{
            print("old value is \(oldValue), new value is \(prenom)")
        }
        willSet{
            print("old value is \(prenom), new value is \(newValue)")
        }
    }

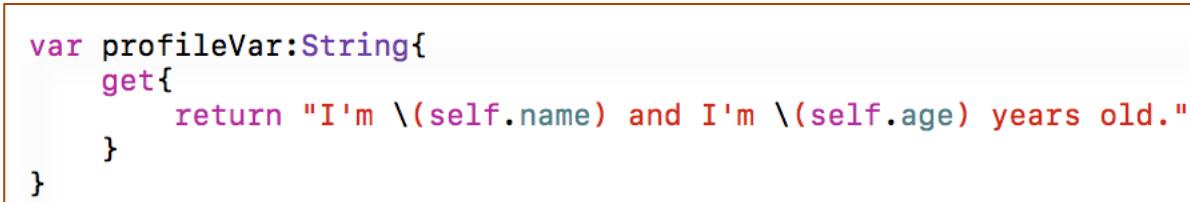
    var prenom:String {
        get {
            return _prenom
        }
        set (aNewValue) {
            if(aNewValue != prenom) {
                _prenom=aNewValue
            }
        }
    }

    func profile()->String{
        return "I'am \(self.name) and I'm \(self.age) years old"
    }

    var profileVar:String{
        get{
            return "I'm \(self.name) and I'm \(self.age) years old."
        }
    }
}

```

Propriété calculée
Prend l'identité d'une personne et affiche/renvoie sa description



```

var p = Person()
p.name = "Matt"
p.age = 40
p.profileVar

```

Swift : Observateurs de l'état d'une propriété

```
class Person1 {  
  
    init(){}
    var name:String="Name"
    var age:Int=0  
  
    private var _prenom:String="Prénom" {
        didSet{
            print("old value is \(oldValue),      value is \(_prenom)")
        }
        willSet{
            print("old value is \(_prenom),      value is \(newValue)")
        }
    }  
  
    var prenom:String {
        get {
            return _prenom
        }
        set (anewValue) {
            if(anewValue != prenom) {
                _prenom=anewValue
            }
        }
    }  
  
    func profile() -> String{
        return "I'am \(self.name) and I'm \(age) years old"
    }
}
```

On peut ainsi effectuer une action lorsqu'une propriété change ou lorsqu'elle vient de changer

didSet code block

willSet code block

Swift : Observateurs de l'état d'une propriété

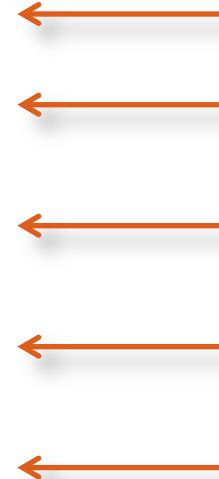
```
class StepCounter {  
    var totalSteps: Int = 0 {  
        willSet(newTotalSteps) {  
            print("About to set totalSteps to \(newTotalSteps)")  
        }  
        didSet {  
            if totalSteps > oldValue {  
                print("Added \(totalSteps - oldValue) steps") // Added 200 steps  
            }  
        }  
    }  
}
```

```
let stepCounter = StepCounter()  
stepCounter.totalSteps = 200 // About to set totalSteps to 200  
stepCounter.totalSteps = 360 // About to set totalSteps to 360  
stepCounter.totalSteps = 896 // About to set totalSteps to 896  
stepCounter.totalSteps = 1432 // Added 536 steps
```

On peut ainsi effectuer une action lorsqu'une propriété change ou lorsqu'elle vient de changer

Swift : lazy et gestion des optionnelles

```
class Person2 {  
    lazy var name:String="Name"  
    var age:Int=0  
    private var _prenom:String?  
  
    lazy var job:String={  
        return "Your job is fun"  
    }()  
  
    init(){}
    var prenom:String {  
        get {  
            if let pren=_prenom {  
                return pren  
            }  
            else {return "Prenom"}  
        }  
        set {  
            _prenom=newValue  
        }  
    }  
  
    func profile()->String{  
        return "I'am \(self.name) and I'm \(self.age) years old"  
    }  
}
```



```
var p2=Person2()  
var prenom=p2.prenom  
print("\(prenom)")  
p2.prenom="Titi"  
p2.job
```

Person2
"Prenom"
"Prenom\n"
Person2
"Your job is fun"

Swift : lazy et gestion des optionnelles

```
class Person2 {  
    lazy var name:String="Name"  
    var age:Int=0  
    private var _prenom:String?  
  
    lazy var job:String={  
        return "Your job is fun"  
    }()  
  
    init(){}
    var prenom:String {  
        get {  
            if let pren=_prenom {  
                return pren  
            }  
            else {return "Prenom"}  
        }  
        set {  
            _prenom=newValue  
        }  
    }  
  
    func profile()->String{  
        return "I'am \(self.name) and I'm \(self.age) years old"  
    }  
}
```

- ← Variable d'instance
- ← Variable d'instance privée optionnelle
- ← fonction
- ← Constructeur
- ← unwrapping

```
var p2=Person2()  
var prenom=p2.prenom  
print("\(prenom)")  
p2.prenom="Titi"  
p2.job
```

Person2
"Prenom"
"Prenom\n"
Person2
"Your job is fun"

Swift : Propriété de type classe

```
class Person {  
    class var species:String{  
        return "Homo sapiens"  
    }  
    var name: String = "Name"  
    var age:Int = 0  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(self.age) years old."  
    }  
}
```

Il s'agit d'une propriété calculée :
Elle retourne un objet

Person.species

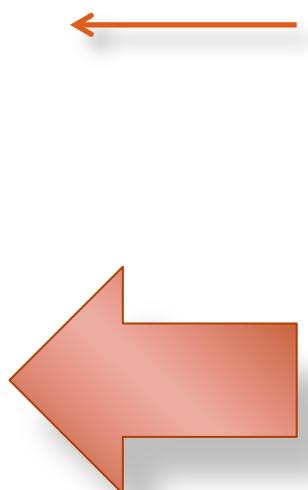
Swift : Méthodes de classe

```
class Person {  
    class var species:String{  
        return "Homo sapiens"  
    }  
  
    class func printDescription() {  
        print("The Person class defines the structure to represent an individual of the  
              species \$(species).")  
    }  
  
    var name: String = "Name"  
    var age:Int = 0  
    func profile() -> String {  
        return "I'm \$(self.name) and I'm \$(self.age) years old."  
    }  
}
```

Au contraire d'une méthode d'instance
les méthodes de classe requièrent la classe
de l'objet : Person.printDescription()

Swift : Héritage et surcharge

```
class Employee:Person2{  
    var employeeNumber:Int?  
    var hourlyRate:Float?  
    var listOfTask=[String]()  
  
    override var prenom:String {  
        get{  
            return "Anonymous"  
        }  
        set {  
            if prenom != newValue{  
                self.prenom=newValue  
            }  
        }  
    }  
  
    func ToDo() -> [String] {  
        listOfTask=["nettoyer", "manger"]  
        return listOfTask  
    }  
  
    override func profile() -> String {  
        return "I'm \(self.prenom) and my hourly rate is \$\(self.hourlyRate)"  
    }  
}
```



Swift : Héritage et surcharge

```
class Employee:Person2{
    var employeeNumber:Int?
    var hourlyRate:Float?
    var listOfTask=[String]()

    override var prenom:String {
        get{
            return "Anonymous"
        }
        set {
            if prenom != newValue{
                self.prenom=newValue
            }
        }
    }

    func ToDo() -> [String] {
        listOfTask=["nettoyer","manger"]
        return listOfTask
    }

    override func profile() -> String {
        return "I'm \(self.prenom) and my hourly rate is \$\(self.hourlyRate)"
    }
}
```

```
var p3=Employee()
p3.profile()
var tab:[String]=p3.todo()

for itask in tab {
    print("\(itask)")
}
tab.append("ranger")
tab.remove(at: 2)
tab
tab.insert("écrire", at:0)
```

Employee
"I'm Anonymous and my hourly rate is \$nil"
["nettoyer", "manger"]

(2 times)

["nettoyer", "manger", "ranger"]
"ranger"
["nettoyer", "manger"]
["écrire", "nettoyer", "manger"]

Swift : Surcharge

```
class Employee: Person {  
    var employeeNumber = 1234567890  
    var hourlyRate = 12.00  
    override func profile() -> String {  
        return "I'm \(self.name) and my hourly rate is \$\(self.hourlyRate)"  
    }  
}
```

```
var p1 = Person()  
p1.name = "Matt"  
p1.lastName = "Campbell"  
p1.age = 40  
print(p1.profile())
```

```
var e1 = Employee()  
e1.name = "Jim"  
e1.lastName = "Smith"  
e1.age = 18  
e1.employeeNumber = 1  
e1.hourlyRate = 15.55  
print(e1.profile())
```

Swift : Surcharge des propriétés

```
class Employee: Person {  
    var employeeNumber = 1234567890  
    var hourlyRate = 12.00  
    override func profile() -> String {  
        return "I'm \(self.name) and my hourly rate is $\(self.hourlyRate)"  
    }  
    override var lastName:String {  
        get {  
            return "Anonymous"  
        }  
        set {  
            _lastName = newValue  
        }  
    }  
}
```

```
var p1 = Person()  
p1.lastName = "Campbell"  
var e1 = Employee()  
e1.lastName = "Smith"  
print(p1.lastName)  
print(e1.lastName)
```

*Campbell
Anonymous*

Swift : Init

```
class Person {  
    var name: String = "Name"  
    var age:Int = 0  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(  
            self.age) years old."  
    }  
}  
  
var p = Person()  
p.name = "Matt"  
p.age = 40
```

Initialisateur par défaut

```
class Person {  
    var name: String  
    var age:Int  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(  
            self.age) years old."  
    }  
    init() {  
        self.name = "Name"  
        self.age = 0  
    }  
}
```

Surcharge de l'initialisateur
par défaut

Swift : Initialisateur personnalisé

```
class Person {  
    var name: String  
    var age:Int  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(self.age) years old."  
    }  
    init() {  
        self.name = "Name"  
        self.age = 0  
    }  
    init(name:String, age:Int) {  
        self.name = name  
        self.age = age  
    }  
}
```

```
var p = Person(name: "Matt", age: 40)  
  
Remarque : déinitialisation  
  
deinit {  
    //Remove any resources outside of standard  
    //types and objects here  
}
```

Swift : Classe dans une classe

```
class Person {  
    var name: String = "Name"  
    var age:Int = 0  
    var health = Health()  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(self.age) years old."  
    }  
}  
  
class Health {  
    var pulse:Int = 100  
    var bmi:Int = 20  
    func profile() -> String {  
        return "Pulse:\(self.pulse), BMI:\(self.bmi)"  
    }  
}
```

```
let p = Person()  
p.health.profile()
```

Swift : Extension

```
class Person {  
    var name: String = "Name"  
    var age:Int = 0  
    func profile() -> String {  
        return "I'm \(self.name) and I'm \(self.age) years old."  
    }  
}  
  
extension Person {  
    var dogYears:Int {  
        get{  
            return self.age * 7  
        }  
    }  
}
```

```
var p = Person()  
p.name = "Matt"  
p.age = 40  
print(p.dogYears)
```

Swift : Extension

Extension : ajouter des méthodes ou des propriétés calculées sur tous les types

```
extension Int {  
  
    var squaredInt:Int{  
        return self*self  
    }  
  
}
```

Int(10).squaredInt

100

Swift : Extension

Extension : on utilise des fonctionnalités avec
des variables existantes

```
extension Rectangle {  
  
    var form:String {  
        return "square"  
    }  
    mutating func Square(){  
        self.height=self.width  
    }  
  
}
```

"square"

Rectangle

```
var rect3=Rectangle(width:640,height:480)  
let form=rect3.form  
print("Le rectangle devient un carré grâce à l'extension")  
rect3.Square()  
rect3.width  
rect3.height
```

Rectangle
"square"
"Le rectangle devient un carré grâce à l'extension"
Rectangle
640
640

Swift : Extension

Extension avec get et set : on utilise les variables existantes

```
extension Person {  
    var anonyme:String {  
        get{  
            return self.name  
        }  
        set(newValue){  
            if(self.name != newValue){  
                self.name=newValue  
            }  
        }  
    }  
}
```

"anonyme"

Person

```
var p4 = Person(name:"toto",age:12)  
p4.anonyme="anonyme"  
let color=p4.anonyme
```

Person

Person

"anonyme"

Swift : Protocole

```
protocol PrinterProtocol {  
    func printThis()  
}
```

```
class aClass:PrinterProtocol {  
    func printThis() {  
        print("Implement printThis for aClass")  
    }  
}
```

```
var obj = aClass()  
obj.printThis()
```

Implement printThis for aClass

Classe parente protocole

```
class aClass: Person, PrinterProtocol{  
    var i = 0  
    func printThis() {  
        print("Implement printThis for aClass")  
    }  
}
```

Swift : Protocole avec variables

```
protocol PrinterProtocol {  
    func printThis()  
    var i:Int {get set}  
}
```

Propriété i requise

```
class aClass:Person, PrinterProtocol {  
    var i = 0  
    func printThis() {  
        println("Implement printThis for aClass & i = \(self.i)")  
    }  
}
```

```
var obj = aClass()  
obj.printThis()
```

Implement printThis for aClass & i = 0

Swift : Délégation Projet

```
class Project {  
    var name = ""  
    var listOfTasks = [Task]()  
}
```

```
var p = Project()  
p.name = "Cook Dinner"  
let taskNames = ["Choose Menu", "Buy Groceries", "Prepare Ingredients", "Cook Food"]
```

```
for name in taskNames{  
    var t = Task()  
    t.name = name  
    p.listOfTasks.append(t)  
}
```

```
class Task {  
    var name = ""  
    var done = false  
}
```

Swift :

Délégation Définition, adoption et implémentation d'un protocole

```
protocol TaskDelegate{  
    func taskStatusHasChanged(task:Task, done:Bool)  
}  
  
class Project:TaskDelegate {  
    var name = ""  
    var listOfTasks = [Task]()  
    func taskStatusHasChanged(task:Task, done:Bool){  
        let status = (task.done ? "DONE" : "IN PROGRESS")  
        print("Task \(task.name) is now \(status)")  
    }  
}  
  
class Task {  
    var name = ""  
    var delegate:TaskDelegate?  
    var done = false }
```

Addition d'une propriété déléguée optionnelle

Swift :

Délégation Définition, adoption et implémentation d'un protocole (suite : appel au délégué; création d'un getter et d'un setter personnalisés)

```
class Task {  
    var name = ""  
    private var _done = false  
    var delegate:TaskDelegate?  
    var done:Bool {  
        get {  
            return _done  
        }  
        set {  
            _done = newValue  
            self.delegate?.taskStatusHasChanged(self, done: _done)  
        }  
    }  
}
```

Swift :

Délégation Définition, adoption et implémentation d'un protocole (suite : assigné la propriété déléguée

```
var p = Project()  
p.name = "Cook Dinner"  
let taskNames = ["Choose Menu", "Buy Groceries", "Prepare Ingredients", "Cook Food"]  
  
for name in taskNames{  
    var t = Task()  
    t.name = name  
    t.delegate = p  
    p.listOfTasks.append(t)  
}  
  
p.listOfTasks[0].done = true  
Task Choose Menu is now DONE
```

III Objective-C et Swift : tableau

```
var myArray: [String] = ["One", "Two", "Three"]
print (myArray[0])
print (myArray[1])
print (myArray[2])
```

```
var myArray: [String] = ["One", "Two", "Three"]
myArray.append("Four")
myArray.append("Five")
myArray.append("Six")
```

```
var myArray: [String] = ["One", "Two", "Three"]
myArray += ["Four", "Five", "Six"]
```

```
var myArray: [String] = ["Two", "Three"]
myArray.insert("One", atIndex: 0)
```

```
var myArray: [String] = ["One", "Two", "Three"]
for myString in myArray {
    print(myString)
}
```

```
var myArray: [String] = ["One", "Two", "Three"]
myArray.removeAtIndex(1)
for myString in myArray {
    print(myString)
}
```

Objective-C et Swift : dictionnaire

```
var person: [String: String] = ["firstName": "John", "lastName": "Doe"]
```

```
print(person["firstName"])
person["firstName"] = "Joe"
```

```
var person: [String: String] = ["firstName": "John", "lastName": "Doe"]
for (myKey, myValue) in person {
    print(myKey + ": " + myValue)
}
```



firstName: John
lastName: Doe

Objective-C et Swift : définition d'une classe

```
Objective-C // In .h interface file
            importStatements
            @interface ClassName : ParentClassName <ProtocolName, ...>
            publicPropertyDeclarations
            publicMethodDeclarations
            @end
            // In .m implementation file
            importStatements
            @implementation ClassName <ProtocolName, ...>
            {
                instanceVariables
            }
            privatePropertyDeclarations
            methodImplementations
            @end
```

```
Swift // In .swift file
      importStatements
      class ClassName: ParentClassName, ProtocolName, ... {
          propertyDefinitions
          methodDefinitions
      }
```

Objective-C et Swift : instantiation d'une classe

```
Objective-C // In MyCustomClass.h
#import Foundation;
static NSString *defaultTitle = @"A Worthy Title";
@interface MyCustomClass : NSObject
@property (copy, nonatomic) NSString *title;
+ (instancetype)instanceWithDefaultTitle;
- (instancetype)initWithTitle:(NSString *)title;
@end
// In MyCustomClass.m
#import "MyCustomClass.h"
@implementation MyCustomClass
+ (instancetype)instanceWithDefaultTitle
{
    return [[MyCustomClass alloc] initWithTitle:nil];
}
- (instancetype)initWithTitle:(NSString *)title
{
    if (self = [super init]) {
        _title = title ?: defaultTitle;
    }
    return self;
}
@end
// In -[SomeOtherClass someMethod] in SomeOtherClass.m
MyCustomClass *myCustomClass1 = [MyCustomClass
instanceWithDefaultTitle]; // myCustomClass1.title =
"A Worthy Title"
MyCustomClass *myCustomClass2 = [[MyCustomClass alloc]
initWithTitle:@"A Great Title"]; // myCustomClass2.title =
"A Great Title"
```

Objective-C et Swift : instantiation d'une classe

```
Swift      // In .swift file
class MyCustomClass {
    class var defaultTitle: String {
        return "A Worthy Title"
    }
    var title: String!
    init(title: String) {
        self.title = title
    }
    convenience init() {
        self.init(title: MyCustomClass.defaultTitle)
    }
}
let myCustomClass1 = MyCustomClass() // myCustomClass1.title =
"A Worthy Title"
let myCustomClass2 = MyCustomClass(title: "A Great Title") // 
myCustomClass2.title = "A Great Title"
```

Swift : exemple 1

```
1 import Foundation
2
3
4 class HelloWorld {
5
6     func logMessage() {
7         let hello = "Hello World!"
8         println(hello)
9     }
10
11 }
```

```
let myHelloWorld = HelloWorld()
myHelloWorld.logMessage()
```

Swift : exemple 2



```
1 // ViewController.swift
2
3
4
5
6
7
8
9 import UIKit
10
11 class ViewController: UIViewController {
12
13     @IBOutlet weak var nameLabel: UILabel!
14
15     @IBAction func showName(sender: AnyObject) {
16         nameLabel.text = "My Name is Brad!"
17     }
18
19     override func viewDidLoad() {
20         super.viewDidLoad()
21         // Do any additional setup after loading the view, typically from a nib.
22     }
23
24     override func didReceiveMemoryWarning() {
25         super.didReceiveMemoryWarning()
26         // Dispose of any resources that can be recreated.
27     }
28
29
30 }
```

Swift : exemple 3

Impossible d'afficher l'image. Votre ordinateur manque peut-être de mémoire pour ouvrir



z. Redémarrez l'ordinateur, puis ouvrez à nouveau le fichier. Si le x rouge est toujours affiché, vous devrez peut-être supprimer l'image avant de la réinsérer.



Impossible d'afficher l'image. Votre ordinateur manque peut-être de mémoire pour ouvrir l'image ou l'image est endommagée. Redémarrez l'ordinateur, puis ouvrez à nouveau le fichier. Si le x rouge est toujours affiché, vous devrez peut-être supprimer l'image avant de la réinsérer.



Impossible d'afficher l'image. Votre ordinateur manque peut-être de mémoire pour ouvrir l'image ou l'image est endommagée. Redémarrez l'ordinateur, puis ouvrez à nouveau le fichier. Si le x rouge est toujours affiché, vous devrez peut-être supprimer l'image avant de la réinsérer.

```
① 37 @IBAction func buttonClick(sender: AnyObject) {  
② 38     stationName.text = myStation.name  
③ 39     stationFrequency.text = String(format: "%.1f", myStation.frequency)  
④ 40  
⑤ 41         if myStation.band() == 1 {  
⑥ 42             stationBand.text = "FM"  
⑦ 43         } else {  
⑧ 44             stationBand.text = "AM"  
⑨ 45         }  
⑩ 46     }  
⑪ 47  
⑫ 48 }  
⑬ 49  
⑭ 50  
⑮ 51 }
```



Impossible d'afficher l'image. Votre ordinateur manque peut-être de mémoire pour ouvrir l'image ou l'image est endommagée. Redémarrez l'ordinateur, puis ouvrez à nouveau le fichier. Si le x rouge est toujours affiché, vous devrez peut-être supprimer l'image avant de la réinsérer.

Swift : exemple 4

```
import Foundation
class Book {
    var title: String = ""
    var author: String = ""
    var description: String = ""

}

9 import UIKit
10
11 class DetailViewController: UIViewController {
12
13
14     var detailItem: AnyObject? {
15         didSet {
16
17             }
18         }
19
20     func configureView() {
21
22         }
23
24     override func viewDidLoad() {
25         super.viewDidLoad()
26         // Do any additional setup after loading the view, typically from a nib.
27         self.configureView()
28     }
29
30     override func didReceiveMemoryWarning() {
31         super.didReceiveMemoryWarning()
32         // Dispose of any resources that can be recreated.
33     }
34
35
36 }
37
```

Swift : exemple 4 (suite)

The screenshot shows the Xcode interface with the storyboard and code editor side-by-side.

Storyboard: The storyboard displays a single view controller titled "Detail". It contains four labels: "Title:", "Author:", "Label", and "Description:". The "Label" and "Description" labels have their "Placeholder" properties set to "Title:" and "Description:" respectively. A blue line connects the "Label" outlet in the storyboard to the "titleLabel" outlet in the code editor.

Code Editor: The code editor shows the implementation of the `DetailViewController` class in Swift. The `titleLabel` outlet is declared as `weak var titleLabel: UILabel!`. The code includes standard boilerplate for `viewDidLoad` and `didReceiveMemoryWarning`.

Connections Inspector: A callout from the storyboard highlights the connection between the "Label" outlet and the `titleLabel` outlet. The Connections Inspector shows the following details:

- Connection: `Outlet`
- Object: `Detail`
- Name: `titleLabel`
- Type: `UILabel`
- Storage: `Weak`

Text Output: Below the code editor, three lines of text are displayed, corresponding to the outlets defined in the storyboard:

```
@IBOutlet weak var titleLabel: UILabel!  
@IBOutlet weak var authorLabel: UILabel!  
@IBOutlet weak var descriptionTextView: UITextView!
```

Swift : exemple 4 (suite)

```
import Foundation

class BookStore {
    var theBookStore: [Book] = []

    init() {
        var newBook = Book()
        newBook.title = "Swift for Absolute Beginners"
        newBook.author = "Bennett and Lees"
        newBook.description = "iOS Programming made easy."
        theBookStore.append(newBook)

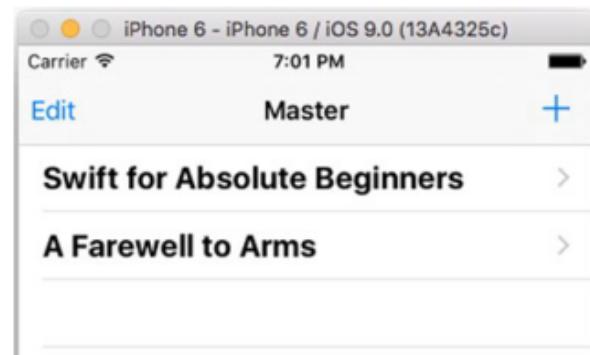
        newBook = Book()
        newBook.title = "A Farewell To Arms"
        newBook.author = "Ernest Hemingway"
        newBook.description = "The story of an affair between an English nurse and an
        American soldier on the Italian front during World War I."

        theBookStore.append(newBook)
    }
}
```

Swift : exemple 4 (suite)

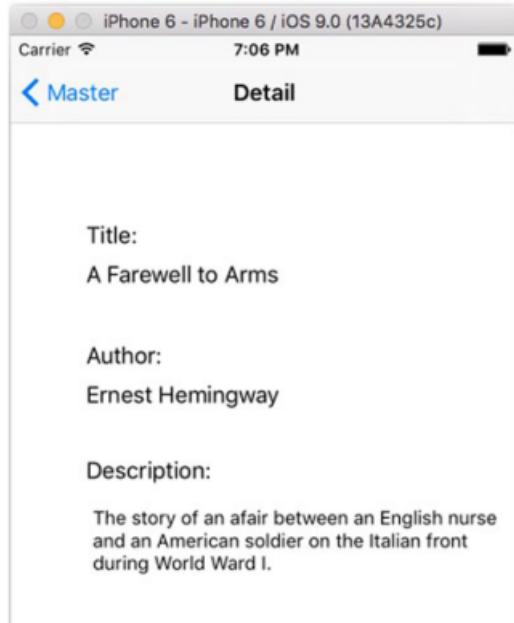
```
//  
//  MasterViewController.swift  
  
import UIKit  
  
  
class MasterViewController: UITableViewController {  
  
    var objects = [AnyObject]()  
    var myBookStore: BookStore = BookStore()  
  
    override func numberOfSectionsInTableView(tableView: UITableView) ->  
        return 1  
    }  
  
    override func tableView(tableView: UITableView, numberOfRowsInSection section: Int)  
-> Int {  
    return myBookStore.theBookStore.count  
}  
  
    override func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath:  
NSIndexPath) -> UITableViewCell {  
    let cell = tableView.dequeueReusableCellWithIdentifier("Cell", forIndexPath:  
indexPath)  
    cell.textLabel!.text = myBookStore.theBookStore[indexPath.row].title  
    cell.accessoryType = UITableViewCellAccessoryType.DisclosureIndicator  
    return cell  
}
```

numberOfSectionsInTableView():
tableView(:numberOfRowsInSection:):
tableView(:cellForRowAtIndexPath:):



Swift : exemple 4 (suite)

```
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {  
    if segue.identifier == "showDetail" {  
        if let indexPath = self.tableView.indexPathForSelectedRow {  
            let selectedBook: Book = myBookStore.theBookStore[indexPath.row]  
            let controller = (segue.destinationViewController as!  
                UINavigationController).topViewController as! DetailViewController  
            controller.detailItem = selectedBook  
  
            controller.navigationItem.leftBarButtonItem = self.splitViewController?.  
                displayModeButtonItem()  
            controller.navigationItem.leftItemsSupplementBackButton = true  
        }  
    }  
}
```



```
func configureView() {  
    if let detail: AnyObject = self.detailItem {  
        var myBook = detail as! Book  
        titleLabel.text = myBook.title  
        authorLabel.text = myBook.author  
        descriptionTextView.text = myBook.description  
    }  
}
```

*Ce que vous devez
(s)avoir avant de
commencer ...*



*Premiers projets:
Hello
Application
interactive*

*Quelques mots
sur objective-C*

*Plus loin...
Autorotation
Photo
Audio, vidéo*

Rappel - synthèse

1 **contrôleur** pour chaque écran
crée la vue, charge le modèle et ajoute
la logique métier

Pour chaque classe, on définit une
- **interface** : décrit le comportement
- **implémentation** : logique interne

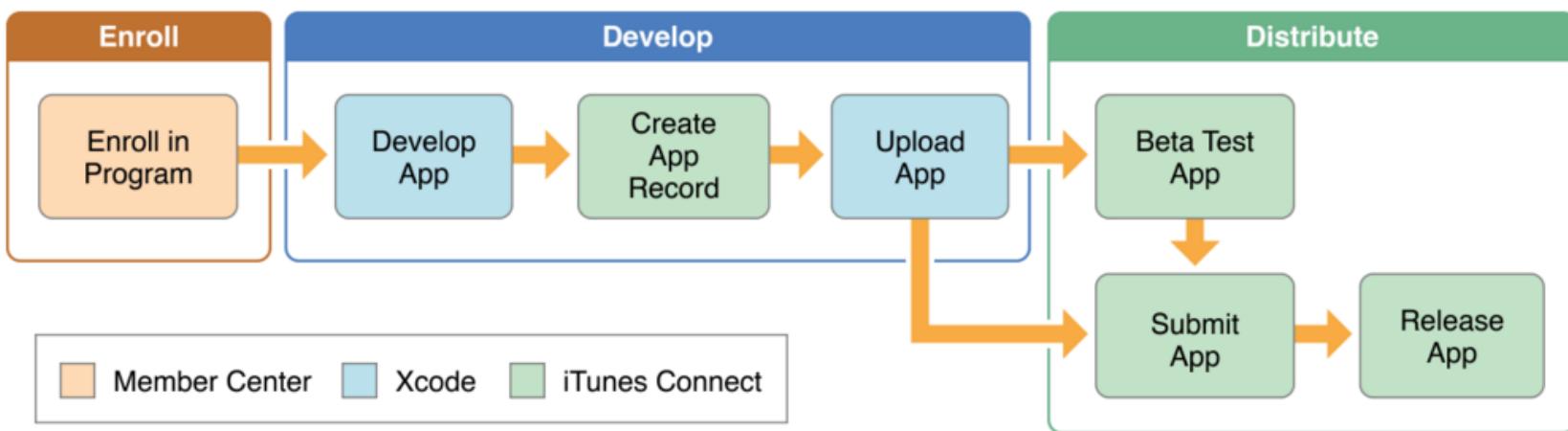
Les méthodes d'une classe peuvent
être des méthodes **d'instance** ou
des méthodes de **classe**.

IBAction : synonyme de void. Permet à IB
de reconnaître cette méthode comme une
Action (Design pattern)

Il est possible de définir des liens
entre les objets ajoutés à la vue et les
propriétés du contrôleur : **IBOutlet**

Il est possible d'enrichir le
comportement d'une classe grâce au
délégué de contrôle

Xcode



Xcode

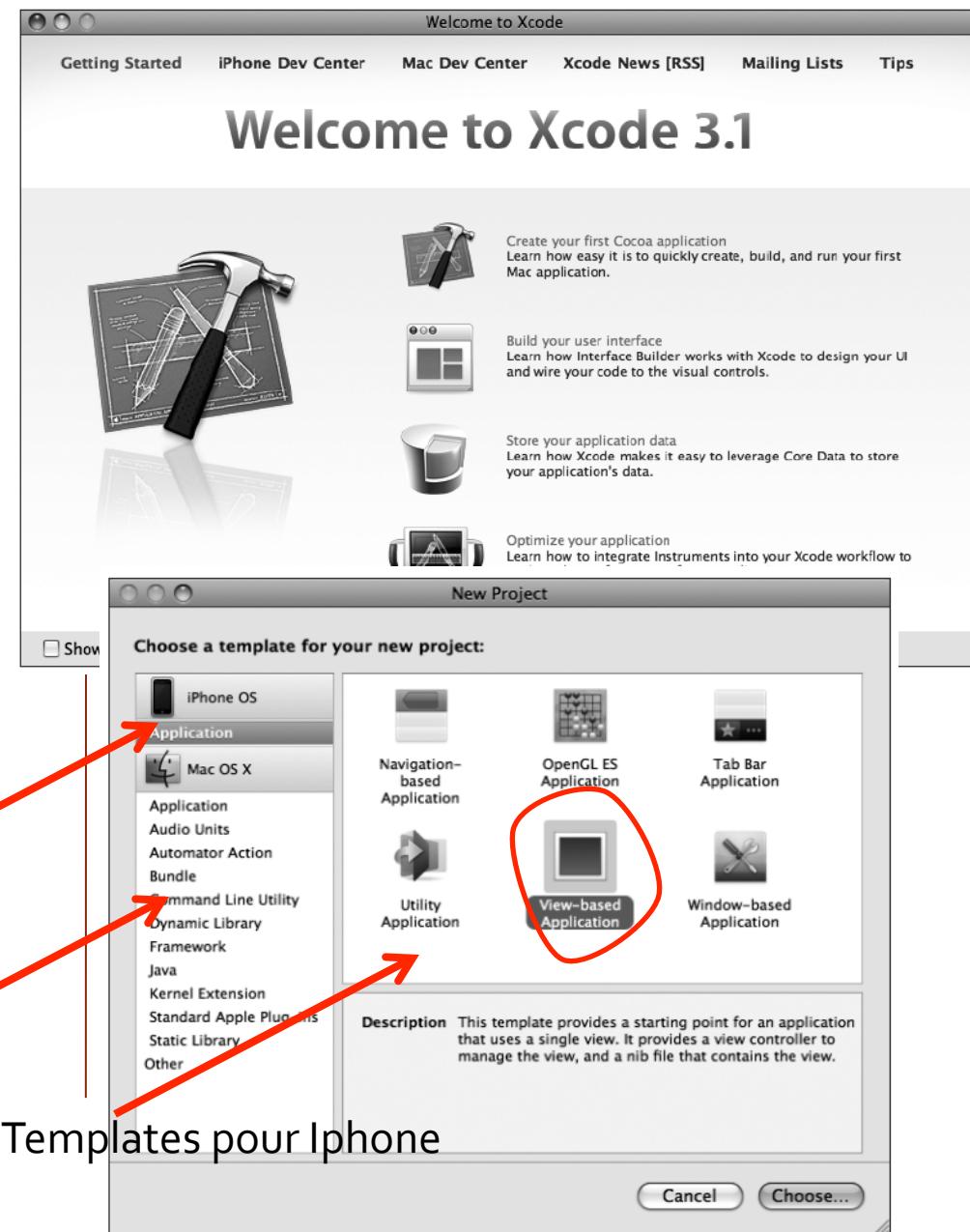
Documentations techniques

iPhone, Mac OS X, tutoriels vidéo,
code ...

Création d'un nouveau projet

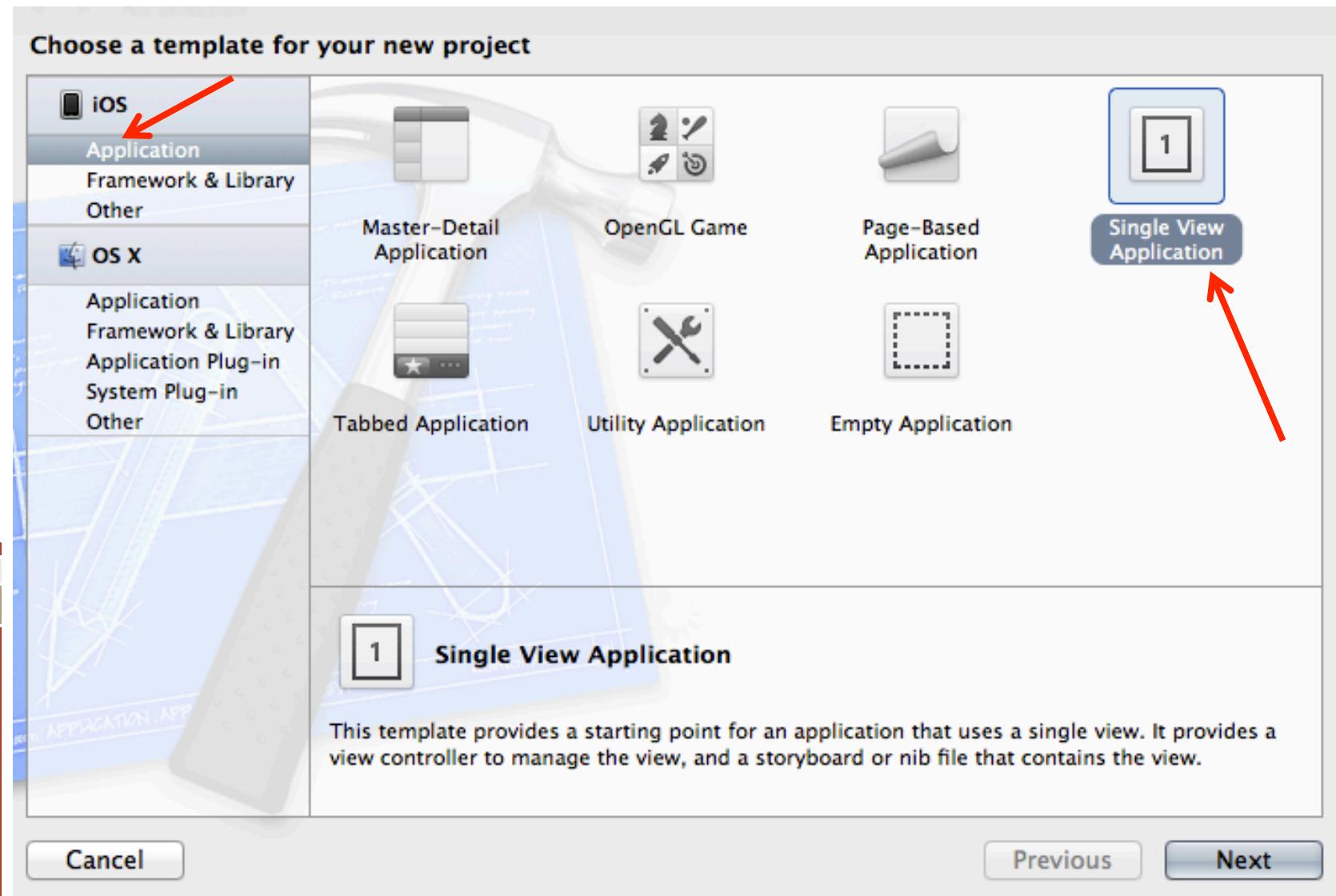
1 seule catégorie de
projet pour Iphone

Catégories pour Mac

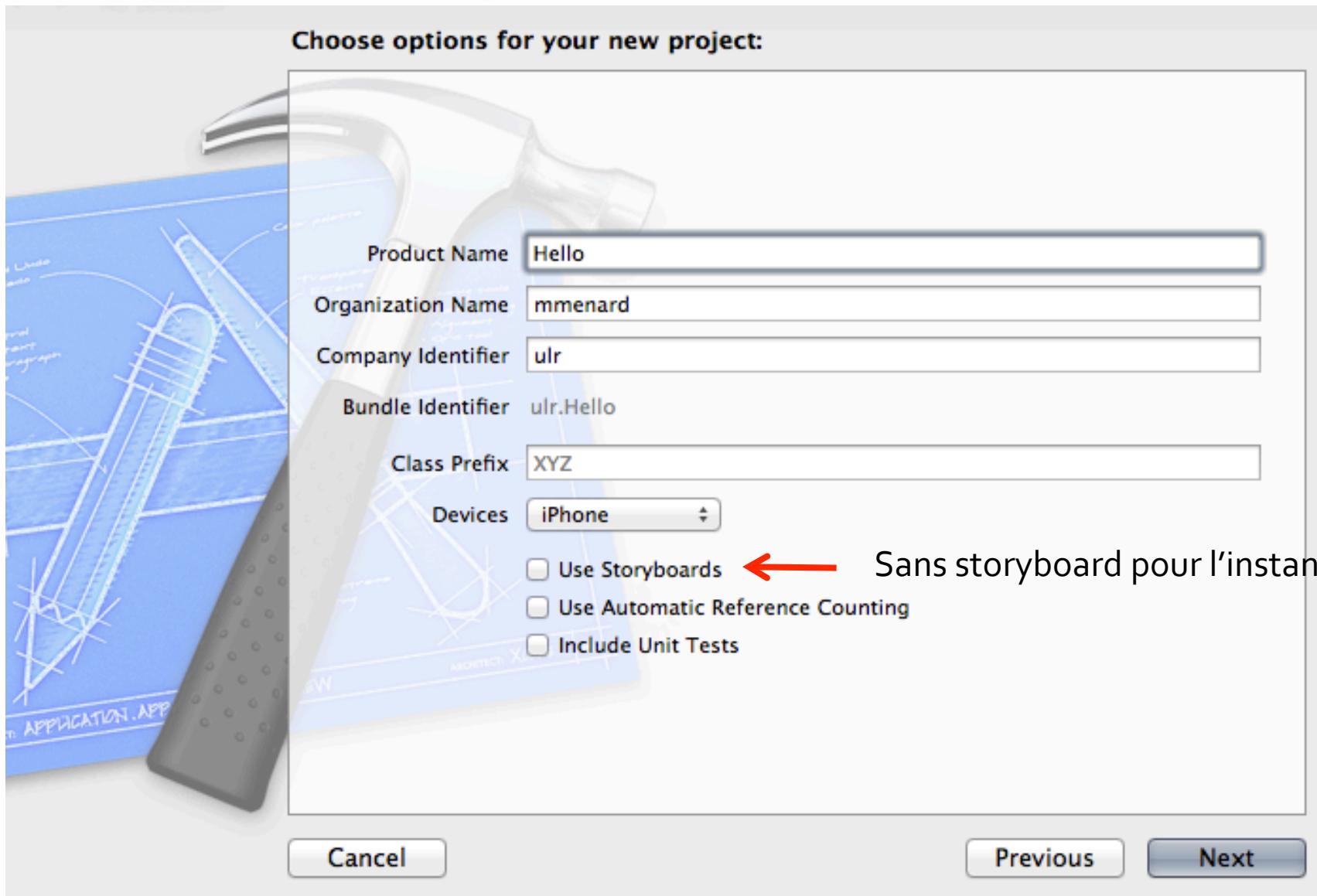


Xcode : 4.x

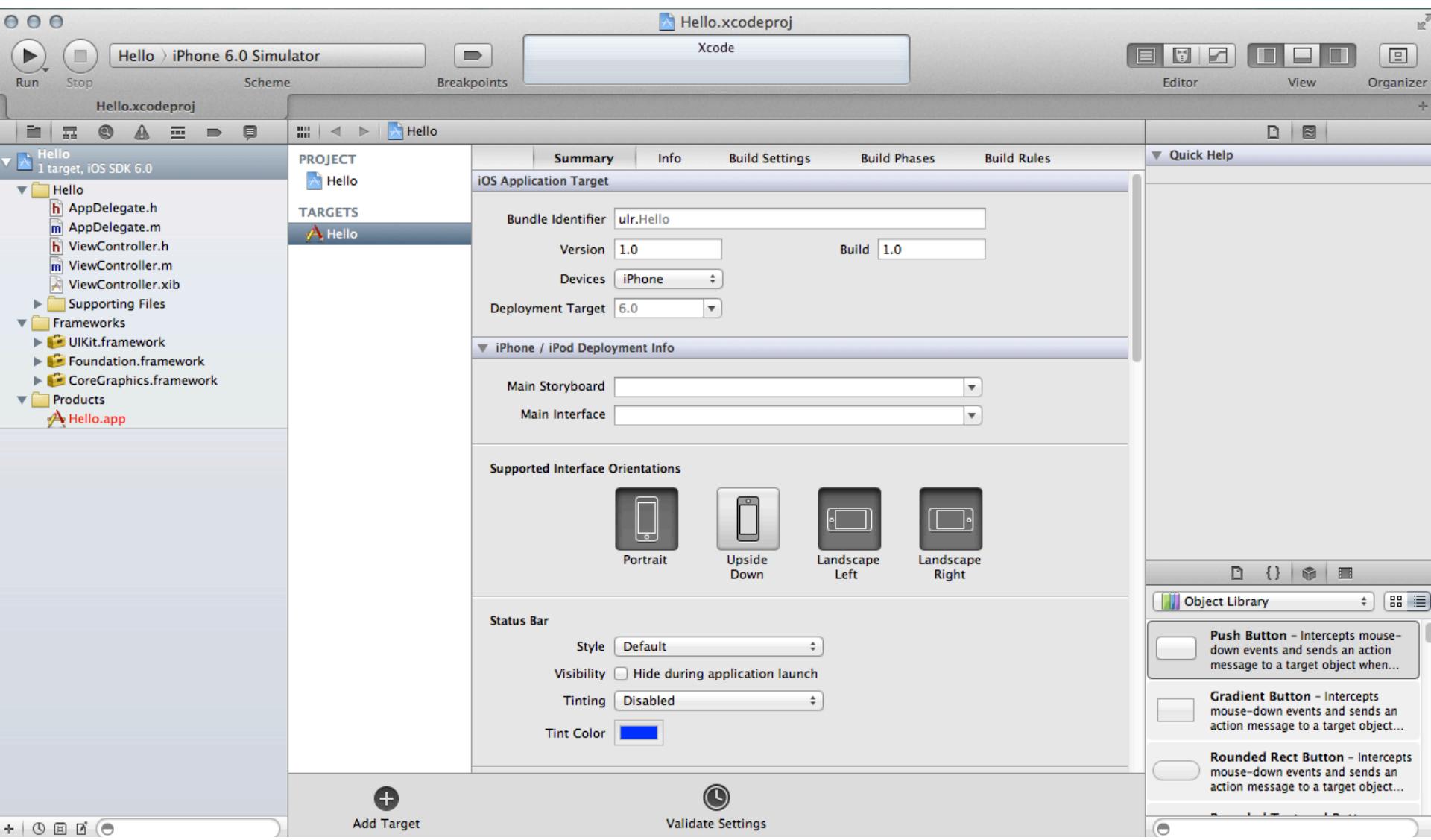
Création d'un nouveau projet



Un nouveau projet



Un nouveau projet



Un nouveau projet

Paramétrage de la compilation

The screenshot shows the Xcode interface with the project 'Hello' selected. The main window displays the 'Build Settings' tab, which is currently active. The settings are organized into sections: 'Architectures', 'Build Active Architecture Only', 'Supported Platforms', 'Valid Architectures', 'Build Locations', and 'Per-configuration Build Products Path'. The 'Build Active Architecture Only' section is expanded, showing 'Debug' set to 'Yes' and 'Release' set to 'No'. The 'Valid Architectures' section lists 'armv7 armv7s'. The 'Build Locations' section shows 'Build Products Path' and 'Intermediate Build Files Path' both set to 'build'. The 'Per-configuration Build Products Path' section is expanded, showing 'Debug' set to 'build/Debug-i...'.

Setting	Resolved	Hello	Hello	iOS 1
Architectures	Standard	Standard	Standard	Standard
Base SDK	Latest iOS (i...)	Latest iOS (i...)	Latest iOS (i...)	iOS 6.0
Build Active Architecture Only	<Multiple val...	<Multiple v...	<Multiple v...	No
Debug	Yes	Yes	Yes	No
Release	No	No	No	No
Supported Platforms	iOS	iOS	iOS	iOS
Valid Architectures	armv7 armv7s	armv7 armv7s	armv7 armv7s	armv7 a
Build Locations				
Build Products Path	build	build	build	build
Intermediate Build Files Path	build	build	build	build
Per-configuration Build Products Path	<Multiple val...	<Multiple val...	<Multip...	<Multip...
Debug	build/Debug-i...	build/Debug-i...	build/Debug-i...	build/D...

Un nouveau projet

PROJECT
Hello

TARGETS
Hello

Ajout d'un framework

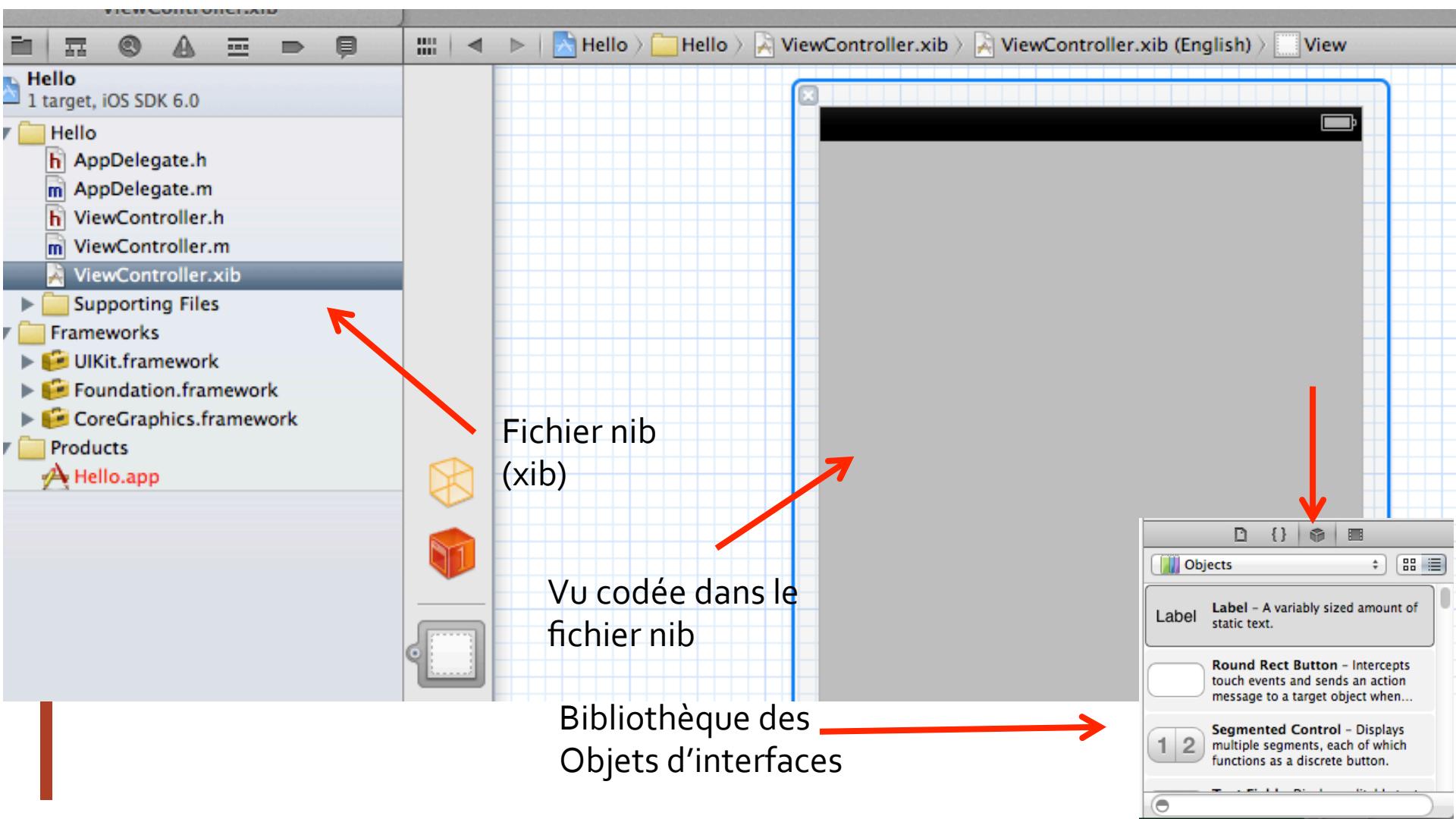
The screenshot shows the Xcode interface with the 'Build Phases' tab selected. On the left, there's a sidebar with 'PROJECT' and 'TARGETS' sections. The 'TARGETS' section is highlighted with a blue bar. In the main area, under 'Build Phases', the 'Link Binary With Libraries' section is expanded, showing three frameworks: 'UIKit.framework', 'Foundation.framework', and 'CoreGraphics.framework', each marked as 'Required'. Below this section is a row with '+' and '-' buttons and the placeholder 'Drag to reorder frameworks'. At the bottom, there's a 'Copy Bundle Resources' section containing five items. A red arrow points from the text 'Ajout d'un framework' to the '+' button in the 'Link Binary With Libraries' section.

Framework	Status
UIKit.framework	Required
Foundation.framework	Required
CoreGraphics.framework	Required

+ - | Drag to reorder frameworks

Copy Bundle Resources (5 items)

Un nouveau projet



Xcode 4

Vue générale de l'éditeur

The screenshot shows the Xcode 4 interface in "General" mode. The top navigation bar displays "photo - photoViewController.h" and "Running photo on iPhone Simulator". The main editor area contains two files: "photoViewController.h" and "photoViewController.m". The left sidebar shows the project structure with targets "photo" and "photoKit.framework", and files like "photoAppDelegate.h", "photoAppDelegate.m", "MainWindow.xib", "photoViewController.h", "photoViewController.m", "photoViewController.xib", "Supporting Files", "photoTests", "Frameworks", and "Products". The bottom status bar shows the date and time (2011-09-20 15:21:23) and the target "photo". A Quick Help panel on the right shows the definition for "m".

```
// photoViewController.h
// photo
//
// Created by Utilisateur on 20/09/11.
// Copyright 2011 __MyCompanyName__. All rights reserved.

#import <UIKit/UIKit.h>
#import <MapKit/MapKit.h>

@interface photoViewController : UIViewController<
    UIImagePickerControllerDelegate, UINavigationControllerDelegate,
    MKMapViewDelegate>
{
    IBOutlet UIButton *takeNewPictureButton;
    IBOutlet UIButton *pickFromCameraRoll;
    IBOutlet UIImageView *imageView;
    IBOutlet UIWebView *myWebView;
    IBOutlet UIWebView *myWebView2;
    IBOutlet UIView *view1;
    IBOutlet UIView *view2;

    IBOutlet MKMapView *carte;
    IBOutlet UILabel *latitudeLabel;
    IBOutlet UILabel *longitudeLabel;
    IBOutlet UILabel *hauteurLabel;
    IBOutlet UILabel *largeurLabel;

    MKCoordinateRegion zone;
}

@property(nonatomic,retain) IBOutlet UIButton *takeNewPictureButton;
@property(nonatomic,retain) IBOutlet UIButton *pickFromCameraRoll;
@property(nonatomic,retain) IBOutlet UIImageView *imageView;
@property(nonatomic,retain) IBOutlet UIWebView *myWebView;
@property(nonatomic,retain) IBOutlet UIWebView *myWebView2;
@property(nonatomic,retain) IBOutlet UIView *view1;
@property(nonatomic,retain) IBOutlet UIView *view2;

@property(nonatomic,retain) IBOutlet MKMapView *carte;
@property(nonatomic,retain) IBOutlet UILabel *latitudeLabel;
@property(nonatomic,retain) IBOutlet UILabel *longitudeLabel;
@property(nonatomic,retain) IBOutlet UILabel *hauteurLabel;
@property(nonatomic,retain) IBOutlet UILabel *largeurLabel;

-(IBAction)getCameraPicture:(id)sender;
-(IBAction)selectExistingPicture:(id)sender;
```

```
// photoViewController.m
// photo
//
// Created by Utilisateur on 20/09/11.
// Copyright 2011 __MyCompanyName__. All rights reserved.

#import "photoViewController.h"

@implementation photoViewController
@synthesize takeNewPictureButton, pickFromCameraRoll,
    imageView, myWebView, myWebView2, view1, view2,
    carte, latitudeLabel, longitudeLabel, hauteurLabel,
    largeurLabel;

- (void) didReceiveMemoryWarning
{
    // Releases the view if it doesn't have a
    // superview.
    [super didReceiveMemoryWarning];

    // Release any cached data, images, etc that
    // aren't in use.
}

#pragma mark - View lifecycle

// Implement viewDidLoad to do additional setup
// after loading the view, typically from a nib.
- (void)viewDidLoad
{
    [super viewDidLoad];
    if (![UIImagePickerController
        isSourceTypeAvailable:
        UIImagePickerControllerSourceTypeCamera]) {
        takeNewPictureButton.hidden=YES;
        pickFromCameraRoll.hidden=YES;
    }

    [myWebView loadRequest:[NSURLRequest
        requestWithURL:[NSURL URLWithString:@"http://www.google.fr"]]];
    //carte.delegate=self;
}

- (void)viewDidUnload
{
```

Local Target Output

photo	2011-09-20 15:21:23.486 photo[1118:e903] Valeur de l'index 1 2011-09-20 15:21:23.488 photo[1118:e903] Valeur de l'index 1
-------	--

Constitution d'un projet

Classes

Les classes Objective-C
Ex : AppDelegate, ViewController
.m, .h

Other Sources

Code sources qui ne sont pas des classes Objective-C
(ex: Hello_World_Prefix.pch, main.m)

Resources

Fichiers ressources : image, son, vidéo,...
dont l'application a besoin. Doit être inclus dans la Sandbox pour l'accès.
Exception : librairie photo et liste d'adresses

ViewController.xib

infos utilisés par Interface Builder

MainWindow.xib

interface principale de l'application (nib file)

HelloWorld-Info.plist

infos sur notre application

Constitution d'un projet (suite)

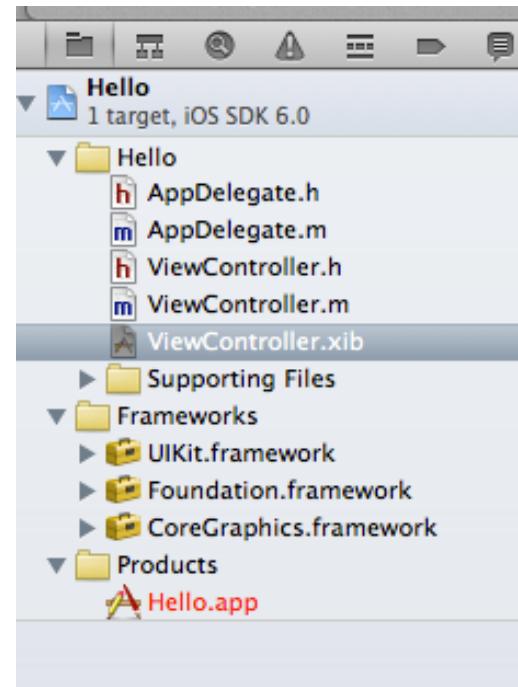
Frameworks

Products

L'organisation précédente ne correspond pas nécessairement à l'organisation sur le disque

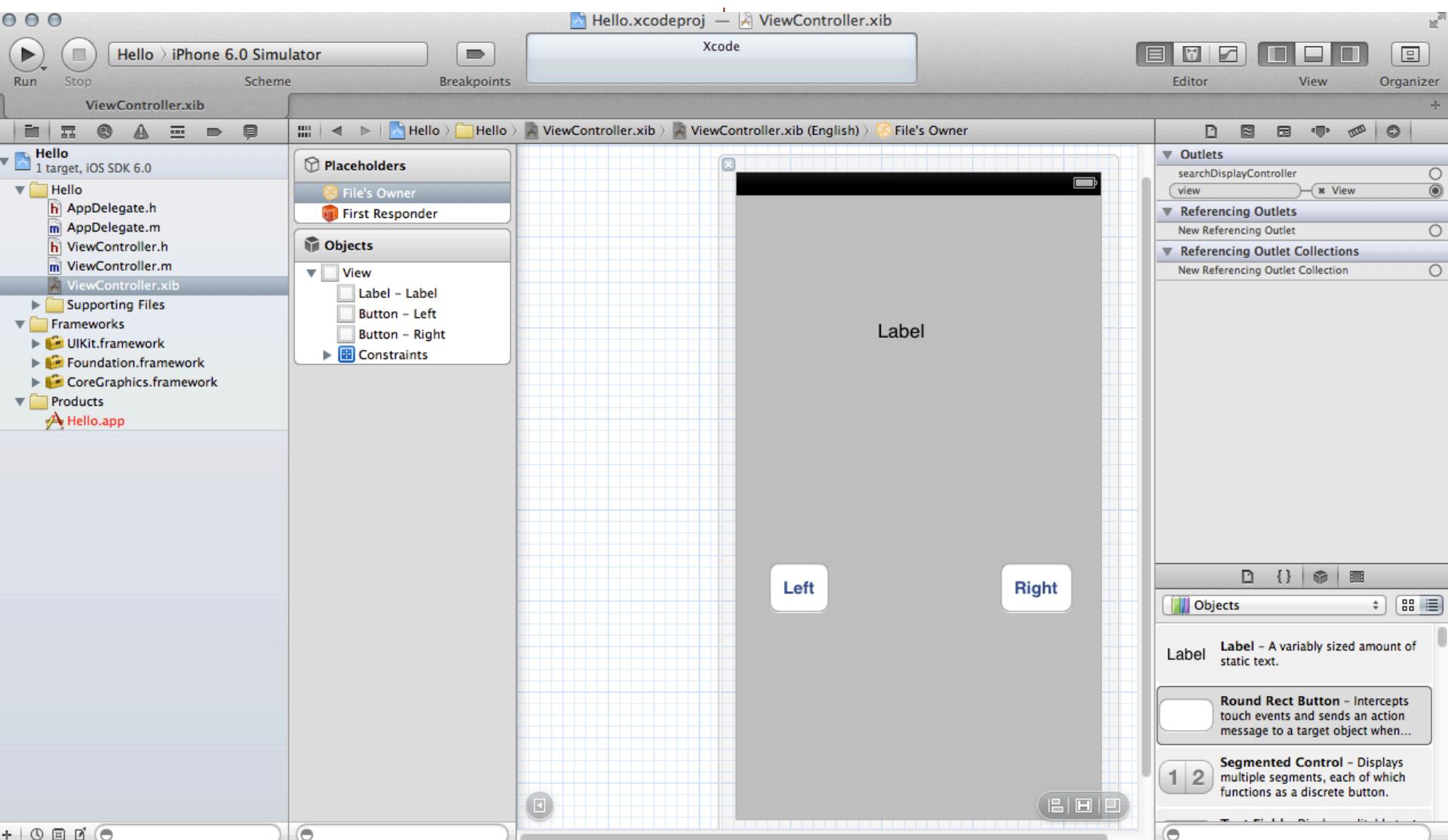
Codes ou ressources supplémentaires
Ex : OpenGL, QuartzCore, CoreGraphics...

Contient l'application que le projet produit
Ex : Hello.app



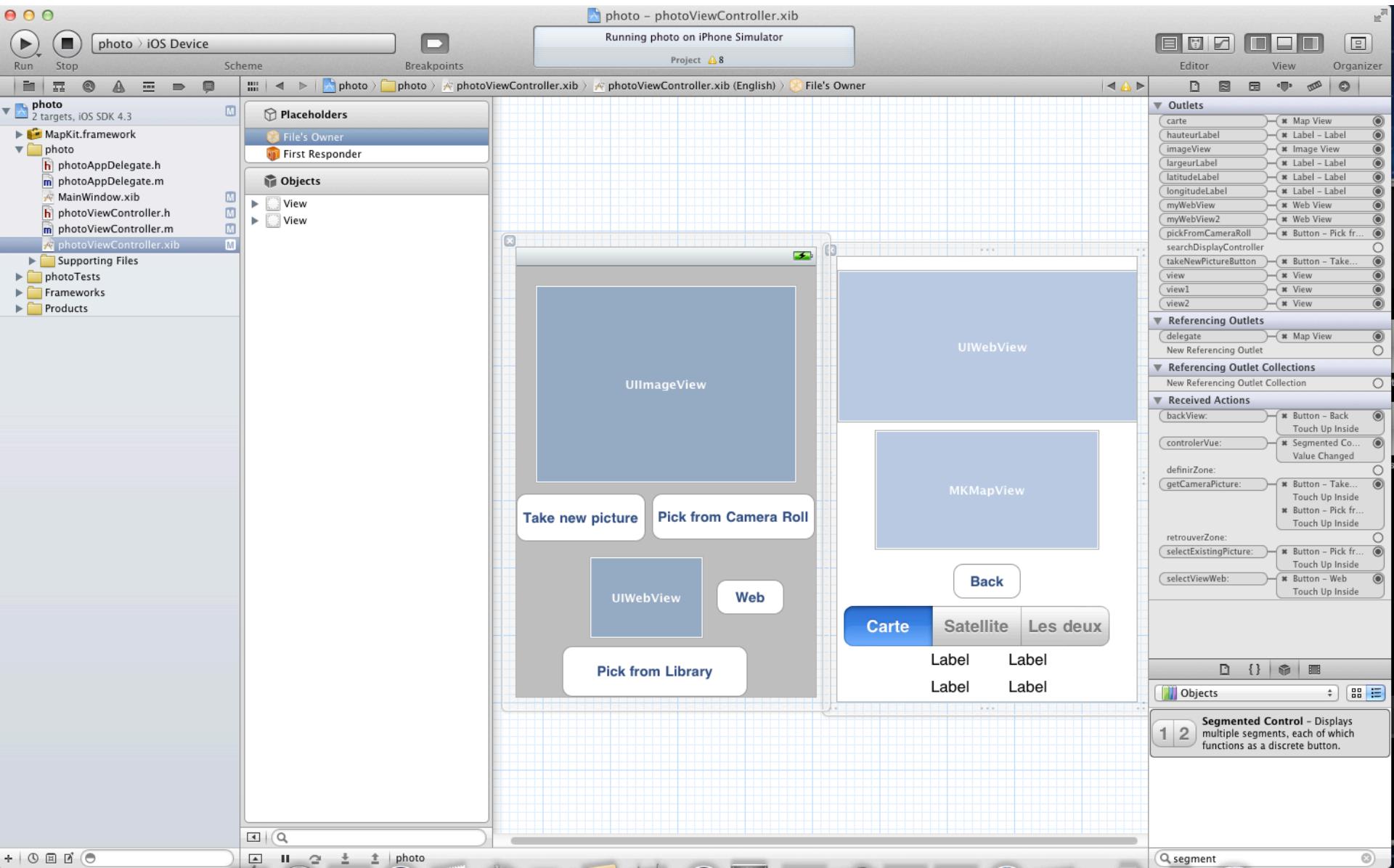
Interface Builder

Interface builder



Interface Builder

Interface builder : exemple de plusieurs vues



Interface Builder

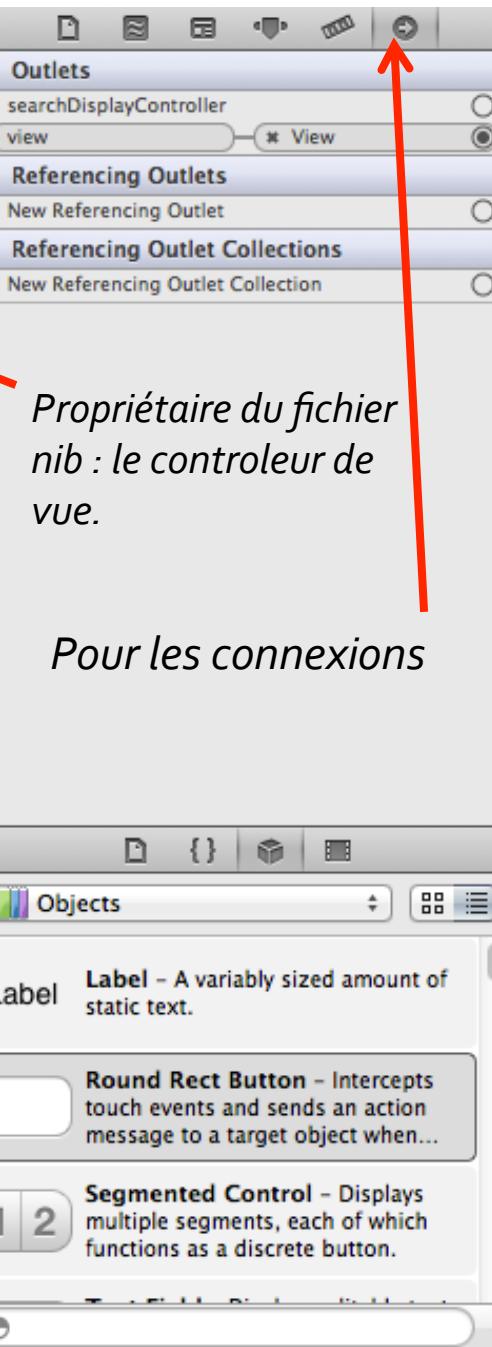
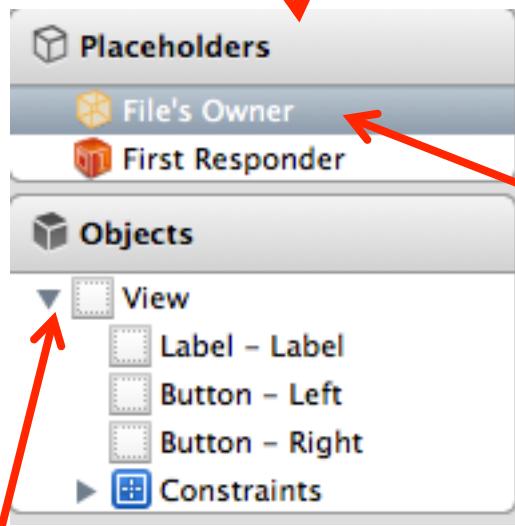
Interface Builder a été créé en 1988.

Utilisé pour développer des applications pour

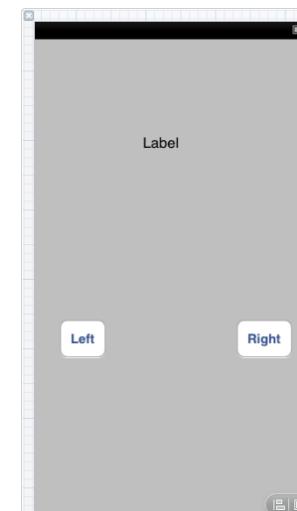
- NextSTEP
- OpenSTEP
- Mac OS X
- Iphone

Les fichiers ont pour extension .xib
(anciennement .nib)

Fenêtre principale nib



La représentation Graphique de View



III Interface Builder

Interface Builder permet par exemple de créer un bouton et spécifier ses attributs (shape, size, label, etc.).

Le bouton sera automatiquement instancié quand votre application se lancera.

Equivalent à

```
UIButton *myButton=[[UIButton alloc] initWithFrame:aRect];
```



File's Owner

Représente l'objet qui a chargé le fichier nib du disque



First Responder

Représente l'objet qui est en permanence interaction avec l'utilisateur (ex:text field)



View

*Représente une instance de la classe UIView
Interagit avec l'utilisateur.*

Interface Builder

Ensemble des objets Cocoa Touch qu'Interface Builder Supporte
(Framework Iphone UIKit)



Objects

- Collection Reusable View** - Defines the attributes and behavior of reusable views in a collection view, ...
- Image View** - Displays a single image, or an animation described by an array of images.
- Text View** - Displays multiple lines of editable text and sends an action message to a target object when...
- Web View** - Displays embedded web content and enables content navigation.
- Map View** - Displays maps and provides an embeddable interface to navigate map content.
- ScrollView** - Provides a mechanism to display content that is larger than the size of the application's window.
- Date Picker** - Displays multiple rotating wheels to allow users to select dates and times.
- Picker View** - Displays a spinning-wheel or slot-machine motif of...

Hello !

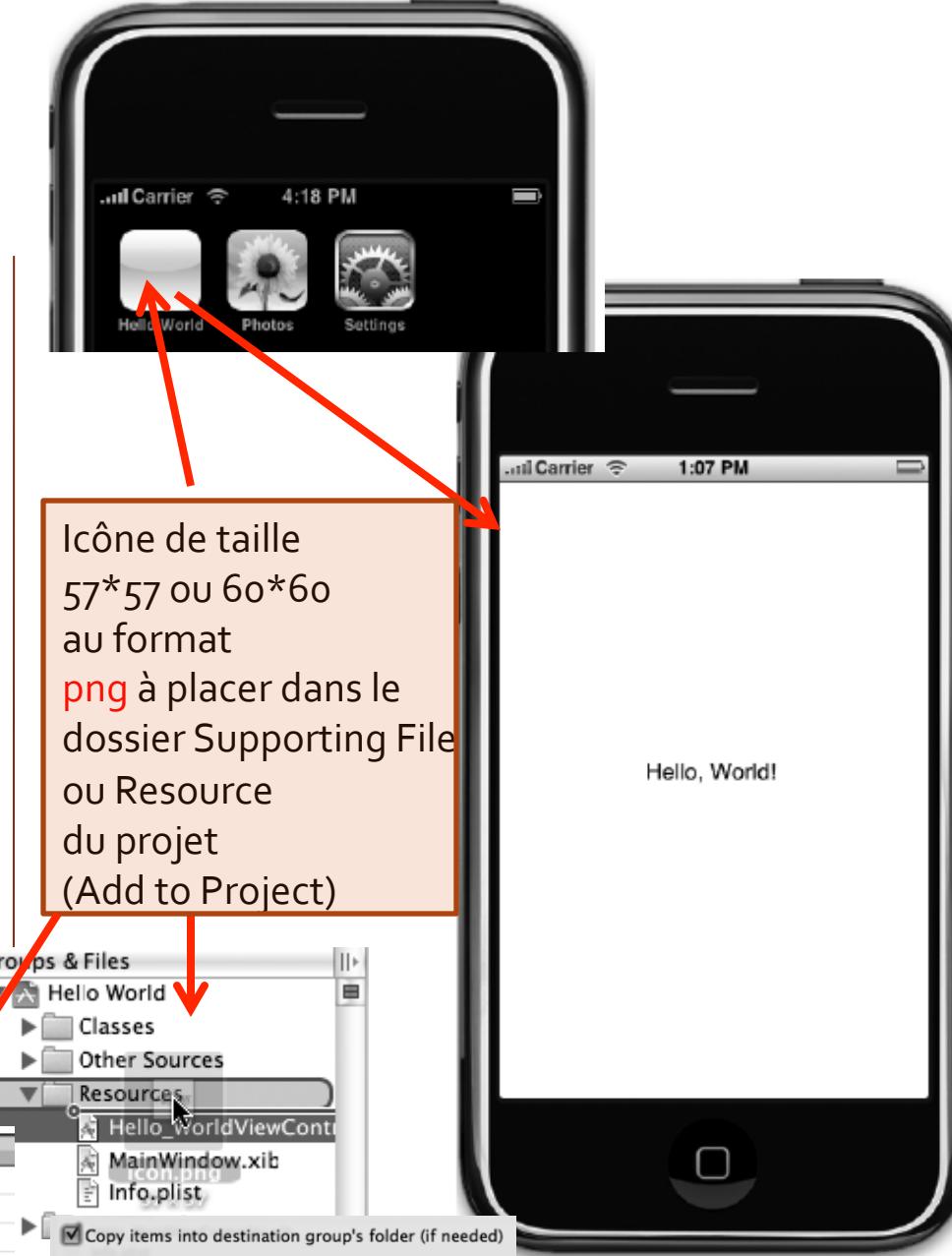
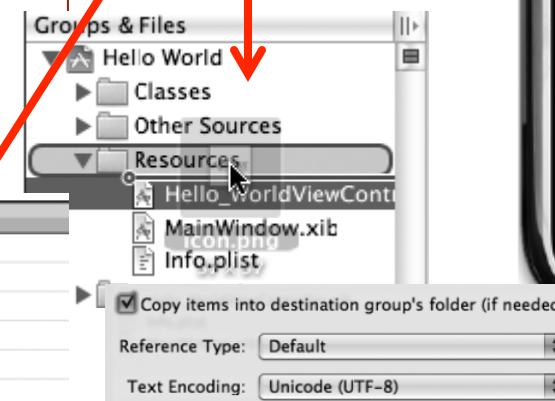
- Sauver votre fichier

-Build / build and Run

Lance la compilation et le programme dans le simulateur.

Dans le dossier Supporting Files ou Resources, sélectionnez Hello-Info.plist

Key	Value
▼ Information Property List	(12 items)
Localization native development re	en
Bundle display name	\${PRODUCT_NAME}
Executable file	\${EXECUTABLE_NAME}
Icon file	



Hello ! (suite)

Vous pouvez modifier l'apparence de votre subview en utilisant l'inspector:

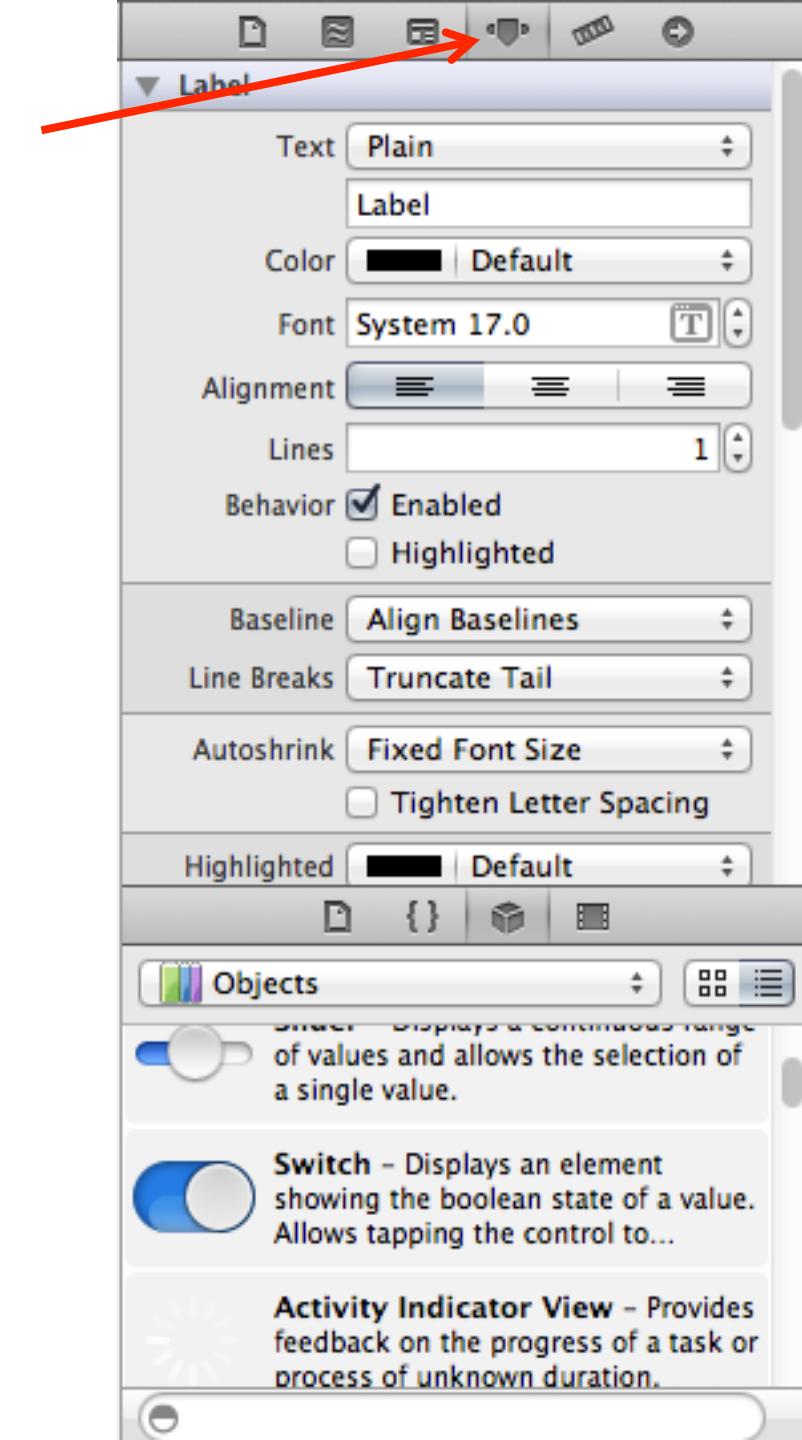
- Selectionnez *Hello World Label*
- *Show the attributes inspector*

Modifiez la taille, la couleur,...

Interface Builder

- crée des objets Objective-C comme vous le feriez dans le code.
- sérialise ces objets dans le fichier nib pour qu'ils puissent être téléchargés en mémoire
- ne nécessite pas de générer du code

Remarque : pour supprimer des applications supprimez le dossier appelé *iPhone Simulator* dans le dossier *Application Support (Home Library)*



||| Une application interactive !

Model-View-Controller Paradigm (MVC)

- **Model** : les classes qui gèrent les données de l'application
- **View** : fenêtres, contrôles, et autres éléments avec qui l'utilisateur peut interagir
- **Controller** : relie les modèles et les vues ensemble. Représente l'application logique qui décide comment gérer les entrées.



Sauvegarder
ou préserver
des données



Interface Builder

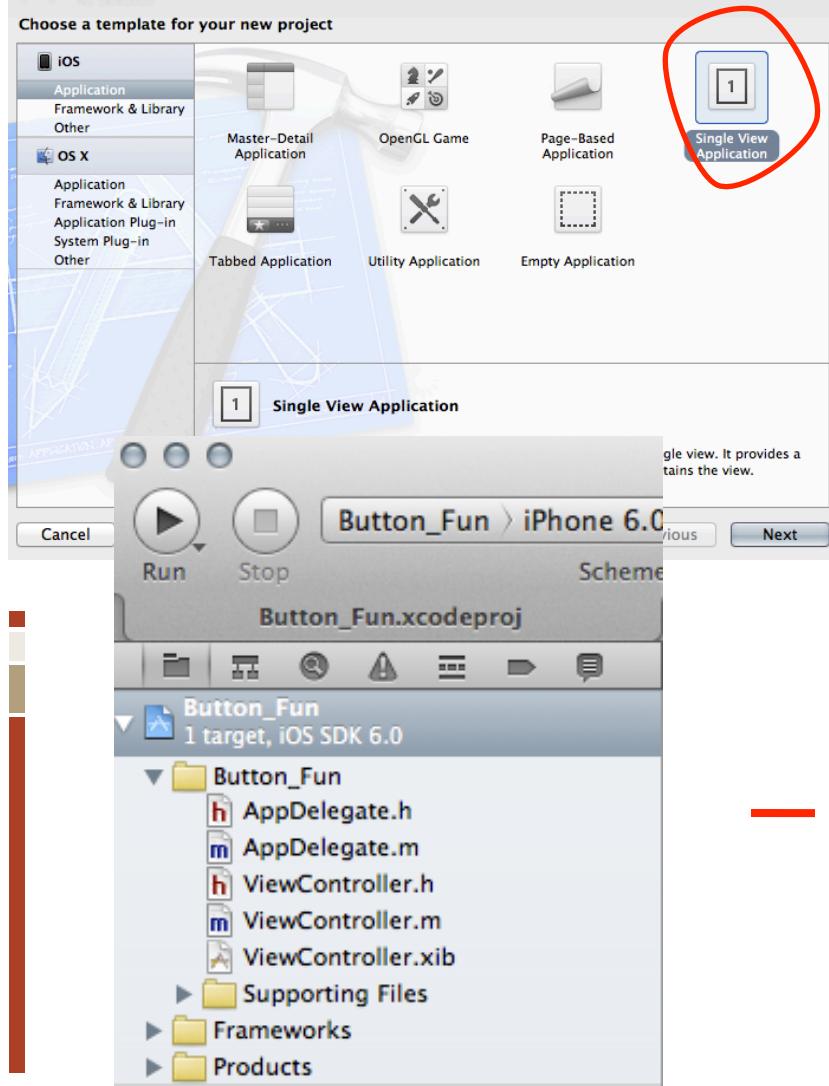


Sous classes du contrôleur
générique existant du
Framework UIViewController



Une application interactive !

(Button Fun)



Deux classes avec .m et .h

1 seule vue

1 classe controller responsable de la gestion de la vue : ViewController

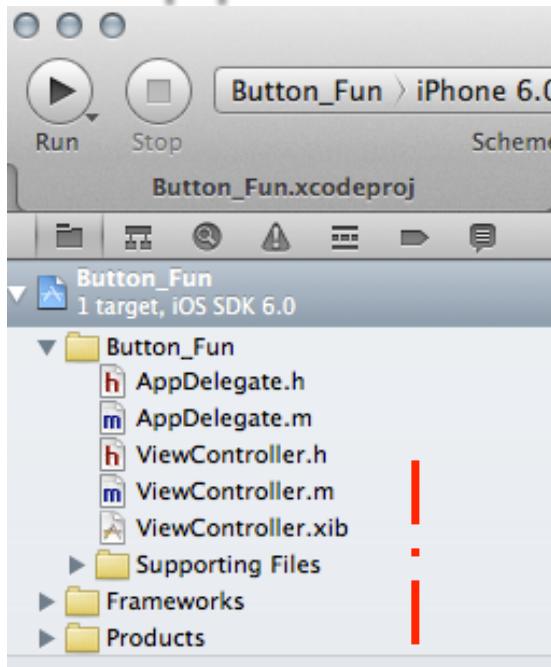
ViewController.h

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController {
}

@end
```

Une application interactive !

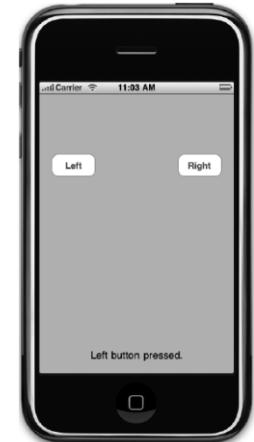


ViewController.h

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController {
}

@end
```



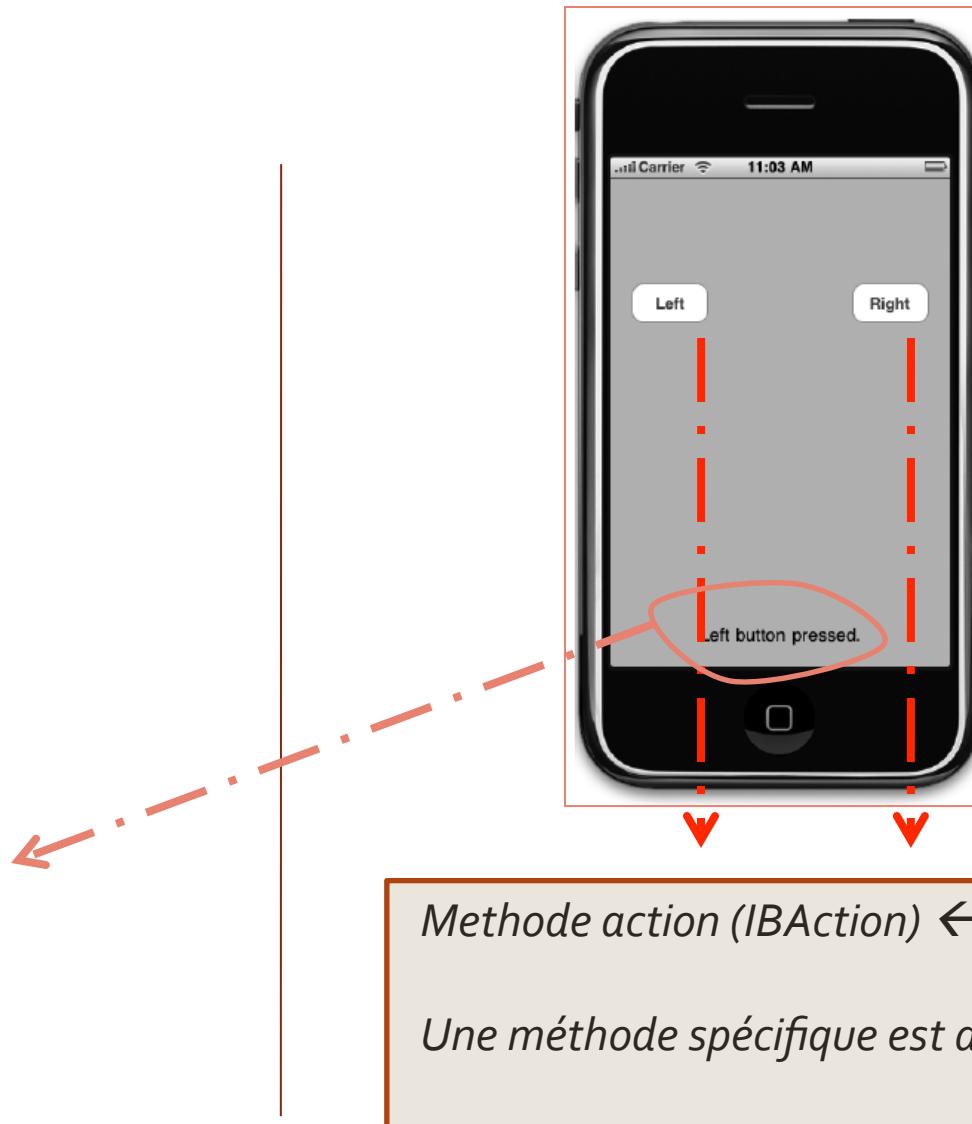
*Notre contrôleur peut faire référence aux objets dans le nib en utilisant un type de variable d'instance nommé **IBOutlet**.*

(pointeur qui pointe vers un objet dans le nib)

IBOutlet → label (permet le changement du texte de label)

/ déclaration de l'IBOutlet et connexion vers l'objet label)

Une application interactive !



Une application interactive !

IBOutlet

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController
{
    IBOutlet UILabel *statusText;
}

@property(nonatomic,retain) IBOutlet
UILabel *statusText;

@end
```

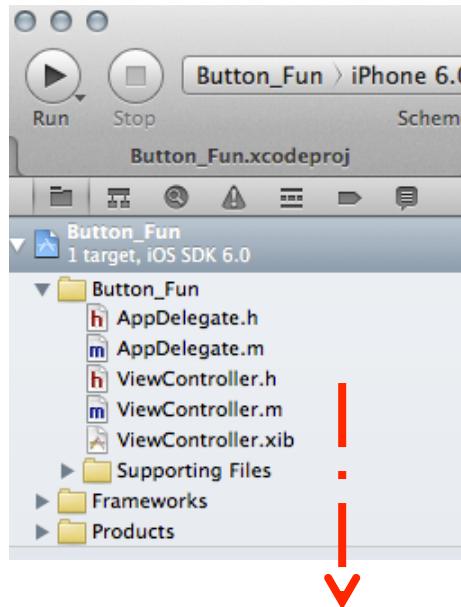
*@property(nonatomic,retain) IBOutlet
UIButton *myButton*

Toute variable d'instance créée et à connecter dans un fichier nib doit être précédée par le Mot clé IBOutlet.

« Cet objet est une variable d'instance que nous allons connecter à un objet dans le nib »

Une application interactive !

IBActions



```
#import <UIKit/UIKit.h>
@interface ViewController : UIViewController {
    IBOutlet UILabel *statusText;
}
@property(nonatomic,retain) IBOutlet UILabel *statusText;
- (IBAction)buttonPressed:(id)sender;
@end
```

Informe les autres classes et IB que nous avons une méthode d'action buttonPressed:(id)sender

Une application interactive !

Property / (getters and setters)

Exemple d'utilisation de property et synthesize

```
@property (nonatomic, retain) id *foo;
```

```
- (id) foo {  
    return foo;  
}  
- (void) setFoo: (id) aFoo {  
    if (aFoo != foo) {  
        [aFoo retain];  
        [foo release];  
        foo = aFoo;  
    }  
}
```

@property

Demande au compilateur de créer les méthodes getter et setter au moment de la compilation.

@property (....,retain)

*Envoie un message « retain » à l'objet auquel nous assignons cette propriété.
/ pour int, float,... inutile.*

@property (nonatomic,...)

Pour les applications iphone

Une application interactive !

Property / (getters and setters)

Exemple d'utilisation de property et synthesize

```
@property (nonatomic, retain) id *foo;
```

```
- (id) foo {  
    return foo;  
}  
  
- (void) setFoo: (id) aFoo {  
    if (aFoo != foo) {  
        [aFoo retain];  
        [foo release];  
        foo = aFoo;  
    }  
}
```

@property

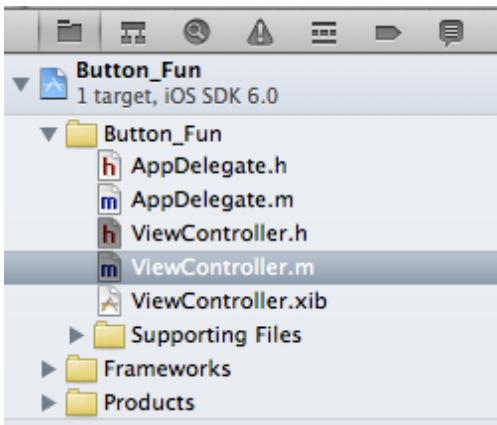
Equivalence entre les écritures :

myVar=[someObject Foo]; accessor
myVar=some.Object.Foo;

[someObject setFoo:myVar]; mutator
someObjects.Foo=myVar;

Une application interactive !

Button_FunViewController.m



Sender : pointe le bouton qui est actionné

UIControlStateNormal : Paramètre qui spécifie le type de contrôle d'état (normal) souhaité.

Deux méthodes invisibles: statusText: et setStatusText:

[statusText setText:newText]

```
@implementation ViewController  
@synthesize statusText;
```

Création automatique des assessor et mutator

```
-(IBAction)buttonPressed:(id)sender  
{
```

```
    NSString *title=[sender titleForState:UIControlStateNormal];  
    NSString *newText=[[NSString alloc] initWithFormat:  
        @">%@ button Pressed",title];  
    statusText.text=newText;  
    [newText release];
```

Implémentation de la méthode d'action

```
}  
- (void)viewDidLoad  
{
```

```
    [super viewDidLoad];
```

// Do any additional setup after loading the view, typically from a nib.

```
}
```

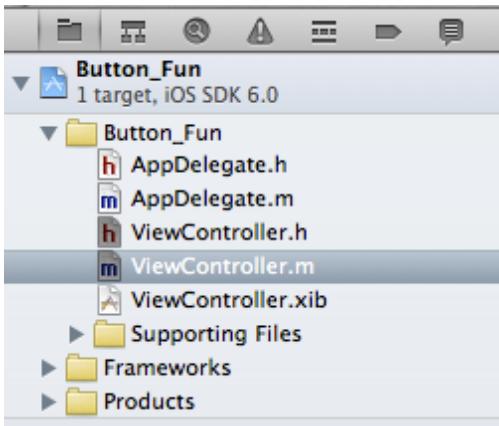
```
- (void)didReceiveMemoryWarning{  
    [super didReceiveMemoryWarning];
```

// Dispose of any resources that can be recreated.

```
@end
```

Une application interactive !

Convenience or factory methods



« si vous ne l'avez pas alloué
ou gardé en mémoire alors
ne le libérez pas !»

```
NSString *newText = [NSString stringWithFormat:  
    @"%@", button.pressed., title];
```

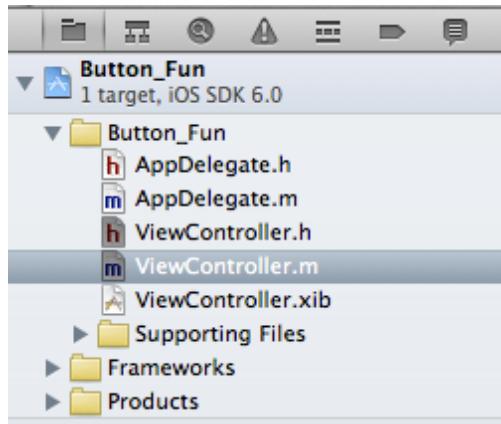
*Retourne un objet autorelease qui ne doit pas être libéré.
Approche non adaptée aux contraintes de l'iphone.*

```
NSString *newText = [[NSString alloc] initWithFormat:  
    @"%@", button.pressed., title];  
  
statusText.text = newText;  
  
[newText release];  
  
[statusText release];
```

Approche adaptée aux contraintes de l'iphone.

Une application interactive !

Allocation cachée!



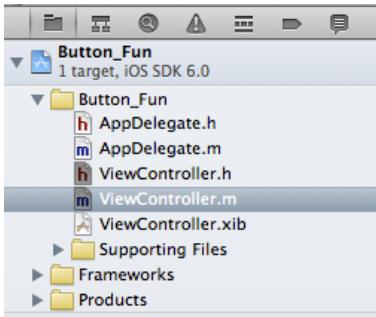
```
NSString *newText = [[NSString alloc] initWithFormat:  
                      @"%@", button.titleLabel.text];  
  
statusText.text = newText;  
  
[newText release];  
  
[statusText release];  
@property (nonatomic, retain) IBOutlet UILabel *statusText;
```

*Implémentation d'une property avec l'option retain pour les outlets.
Il est de notre responsabilité de les libérer dans dealloc !*

Interface Builder : mutator → assigner l'outlet via retain

Une application interactive !

Ecriture condensée/imbriquée



```
NSString *title = [sender titleForState:UIControlStateNormal];
NSString *newText = [[NSString alloc] initWithFormat:
                     @"%@ button pressed.", title];
statusText.text = newText;
[newText release];
```

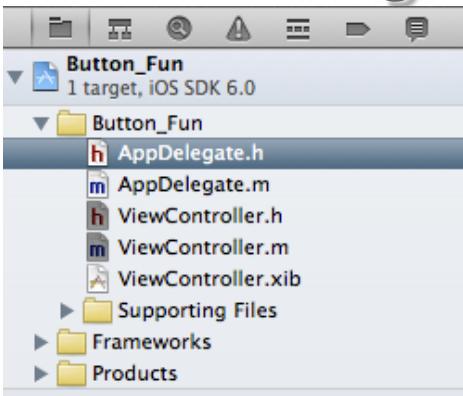


Gestion de la mémoire différente

```
statusText.text = [NSString stringWithFormat:@"%@ button pressed.",
                   [sender titleForState:UIControlStateNormal]];
```

Une application interactive !

Classe déléguée



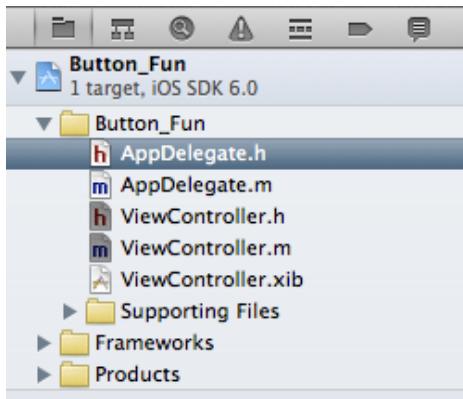
*Très utilisée dans COCOA Touch :
En charge de certaines tâches au profit d'une autre classe
Délégué de l'instance de UIApplication*

Chaque application Iphone a une et une seule instance de UIApplication responsable

- de la boucle de lancement,
- de la gestion niveau application : routage des entrées vers la classe contrôleur appropriée,
- de l'appel à des méthodes déléguées si elles existent

Une application interactive !

Classe déléguée



Cette classe doit être conforme au protocole appelé **UIApplicationDelegate**.

UIApplicationDelegate Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/UIKit.framework
Availability	Available in iPhone OS 2.0 and later.

```
#import <UIKit/UIKit.h>

@class ViewController;

@interface AppDelegate : UIResponder <UIApplicationDelegate>
@property (strong, nonatomic) UIWindow *window;
@property (strong, nonatomic) ViewController *viewController;
@end
```



Voir
Documentation
du protocole

Une application interactive !

Classe déléguée

```
#import "AppDelegate.h"
#import "ViewController.h"

@implementation AppDelegate
- (void)dealloc
{
    [_window release];
    [_viewController release];
    [super dealloc];
}

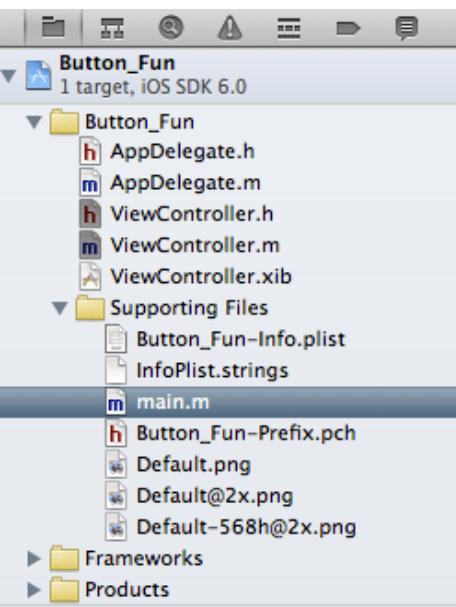
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    self.window = [[[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]] autorelease];
    // Override point for customization after application launch.
    self.viewController = [[[ViewController alloc] initWithNibName:@"ViewController" bundle:nil] autorelease];
    self.window.rootViewController = self.viewController;
    [self.window makeKeyAndVisible];
    return YES;
}
@end
```

Dès que l'application est lancée (setup) et est prête à interagir avec l'utilisateur, cette méthode de la classe déléguée se lance.

Additionne notre vue controller's view comme une sous-vue(subview) à la fenêtre principale et rend la fenêtre visible.

Une application interactive !

Edition de main.m



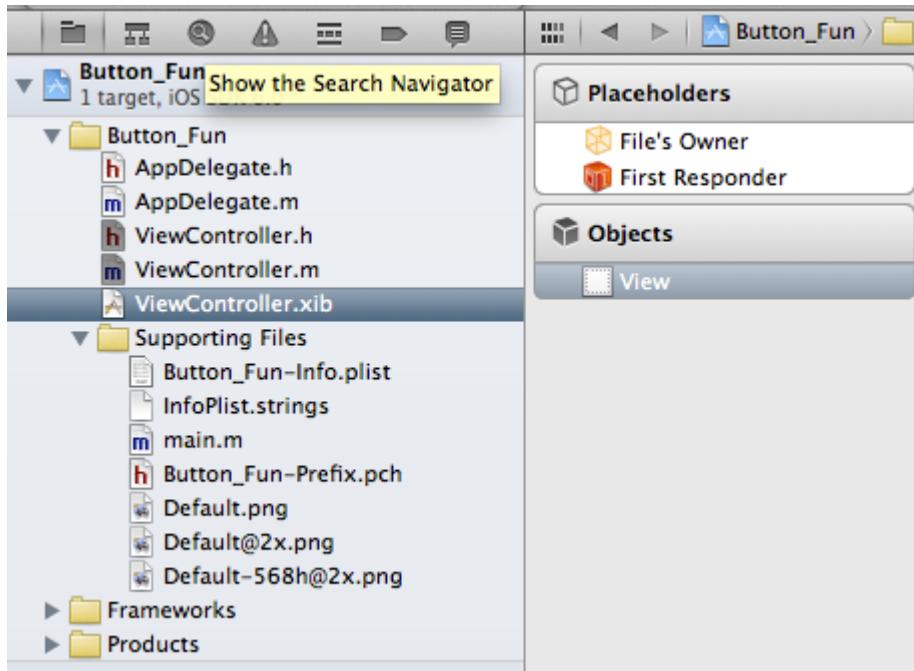
Button_Fun-Info.plist (ex : icône)
ViewController.xib (ex :label « hello.World »)
main.m : création de l’instance de UIApplication,
de la boucle d’événement et du bassin d’autorelease.

```
#import <UIKit/UIKit.h>
#import "AppDelegate.h"

int main(int argc, char *argv[])
{
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([AppDelegate class]));
    }
}
```

Une application interactive !

Edition de MainWindow.xib

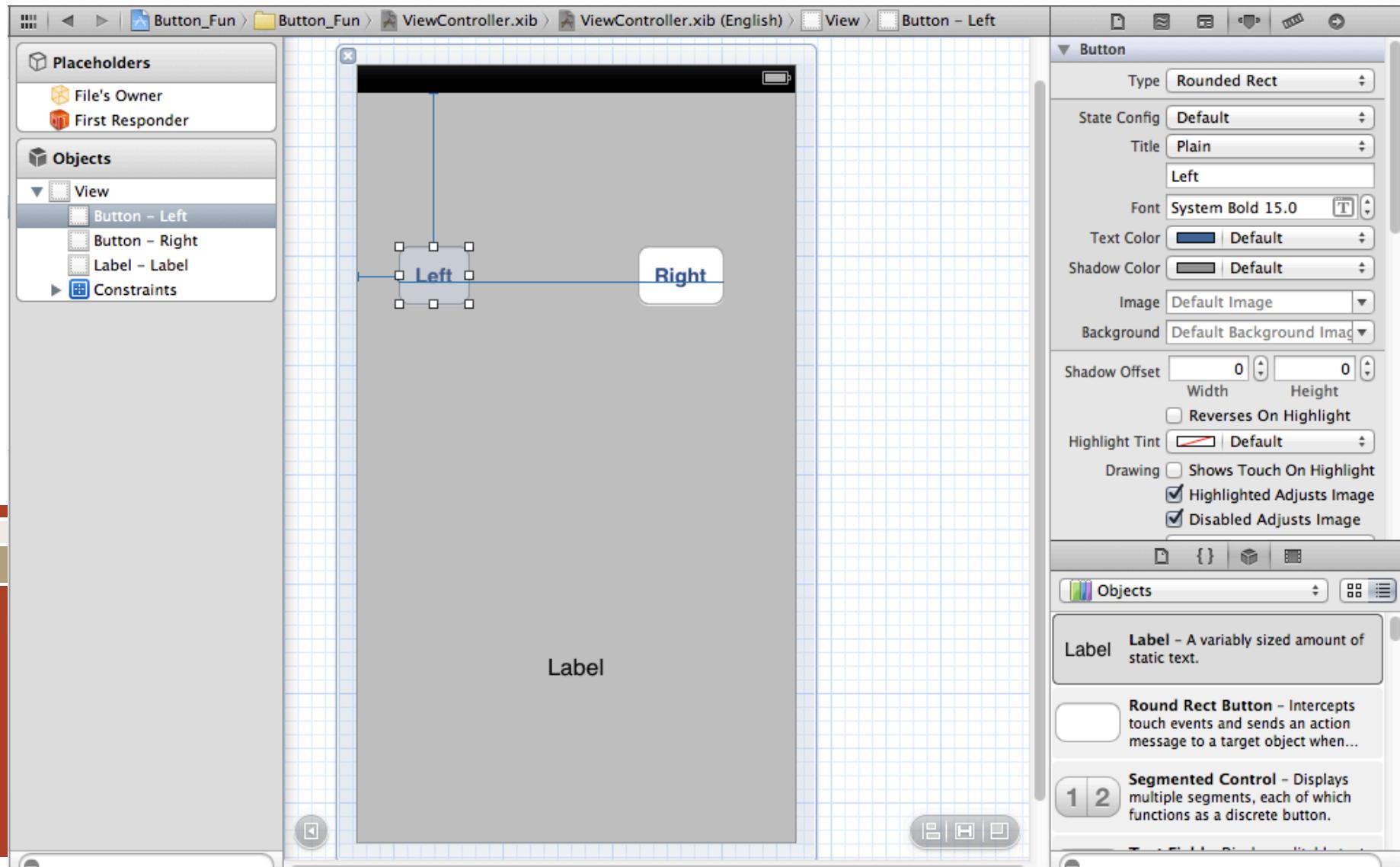


Instance de ViewController.

Instance de la vue de ViewController
(ex : viewController.view)

Une application interactive !

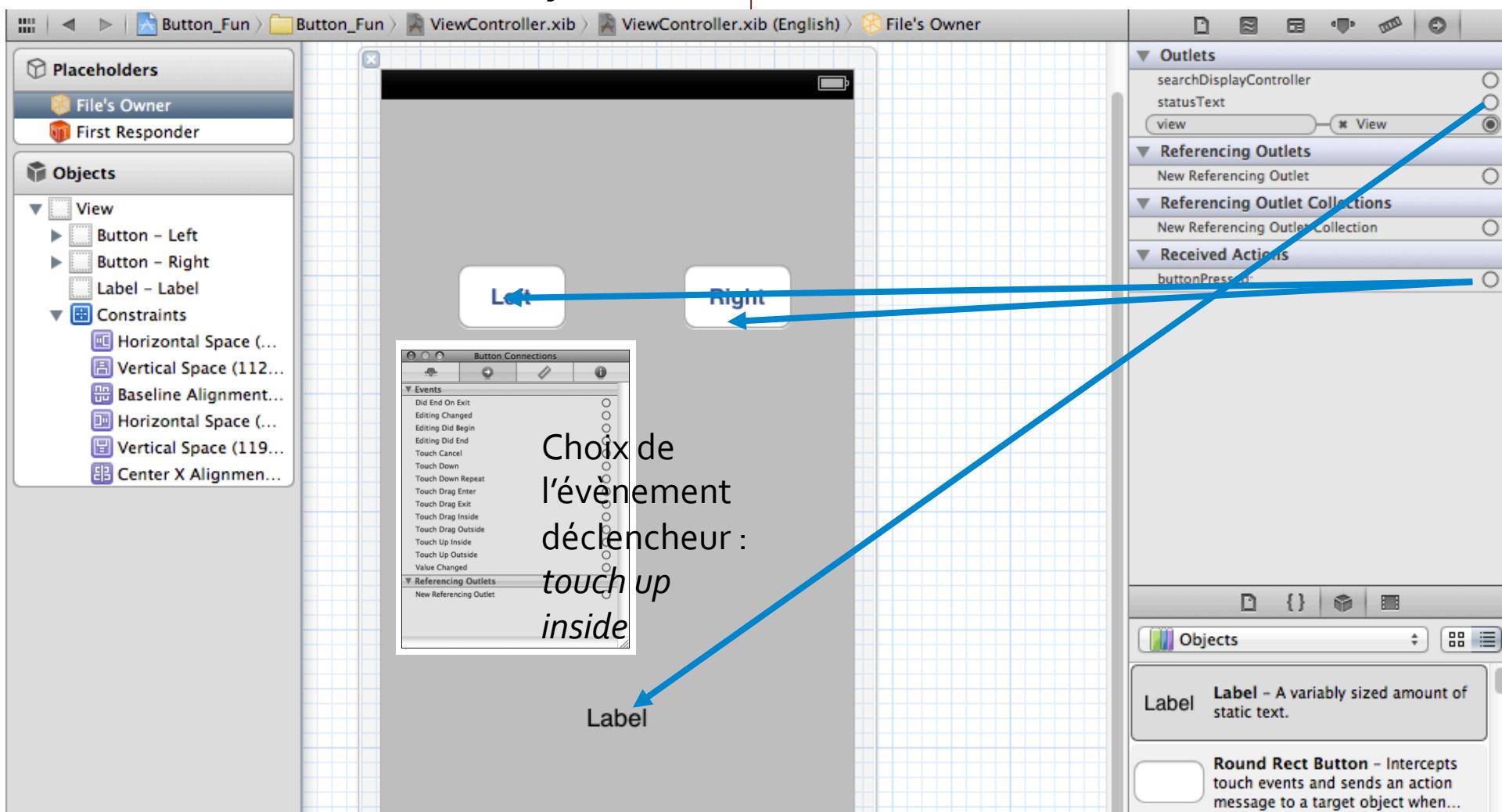
Edition de ViewController.xib



Une application interactive !

Connecting people !

Lorsque nous appuierons sur Left ou Right, la méthode d'action *buttonPressed* sera déclenchée
La VI *statusText* est connecté à l'objet label de l'interface

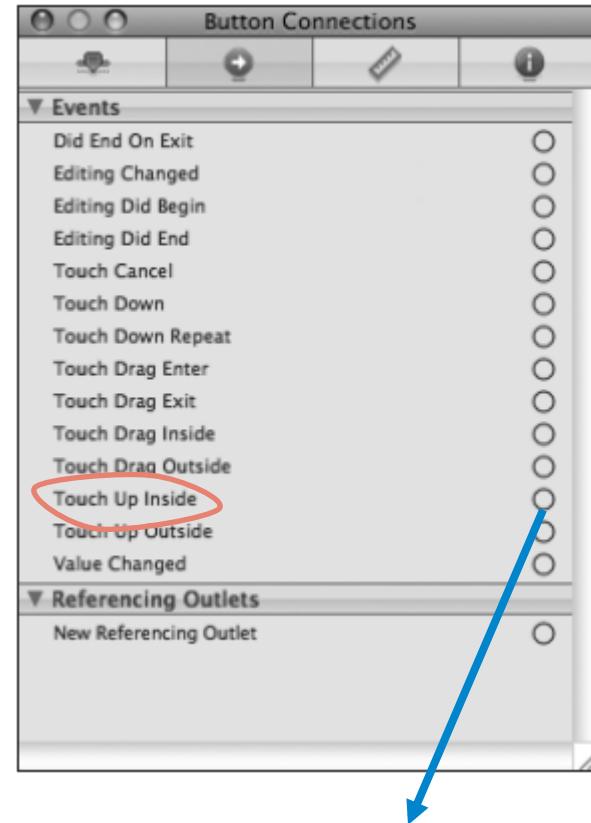


Une application interactive !

Connecting people !

Lorsque l'utilisateur appuie sur le bouton, nous voulons que la méthode buttonPressed : soit appelée.

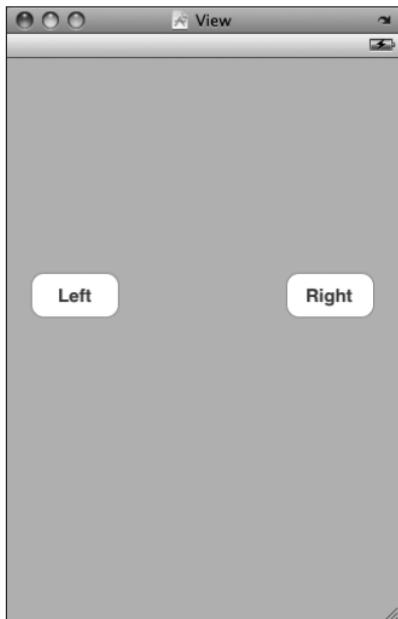
« au moment de lever le doigt, nous devons être sur le bouton »



Sélectionnez buttonPressed:

Une application interactive !

Connecting people !



File's Owner

- *représente l'instance exclusive de l'application de la classe ViewController.*

Événement bouton → l'icône File'sOwner

- *Nous demandons à IB d'appeler la méthode sélectionnée quand l'évènement apparaît.*

Utilisateur → bouton

- La méthode buttonPressed: de la classe ViewController sera appelée.

Une application interactive !

Build and Run !

Introduction au MVC

Création et connexion d'outlets et d'actions

Implémentation des contrôleurs de vues

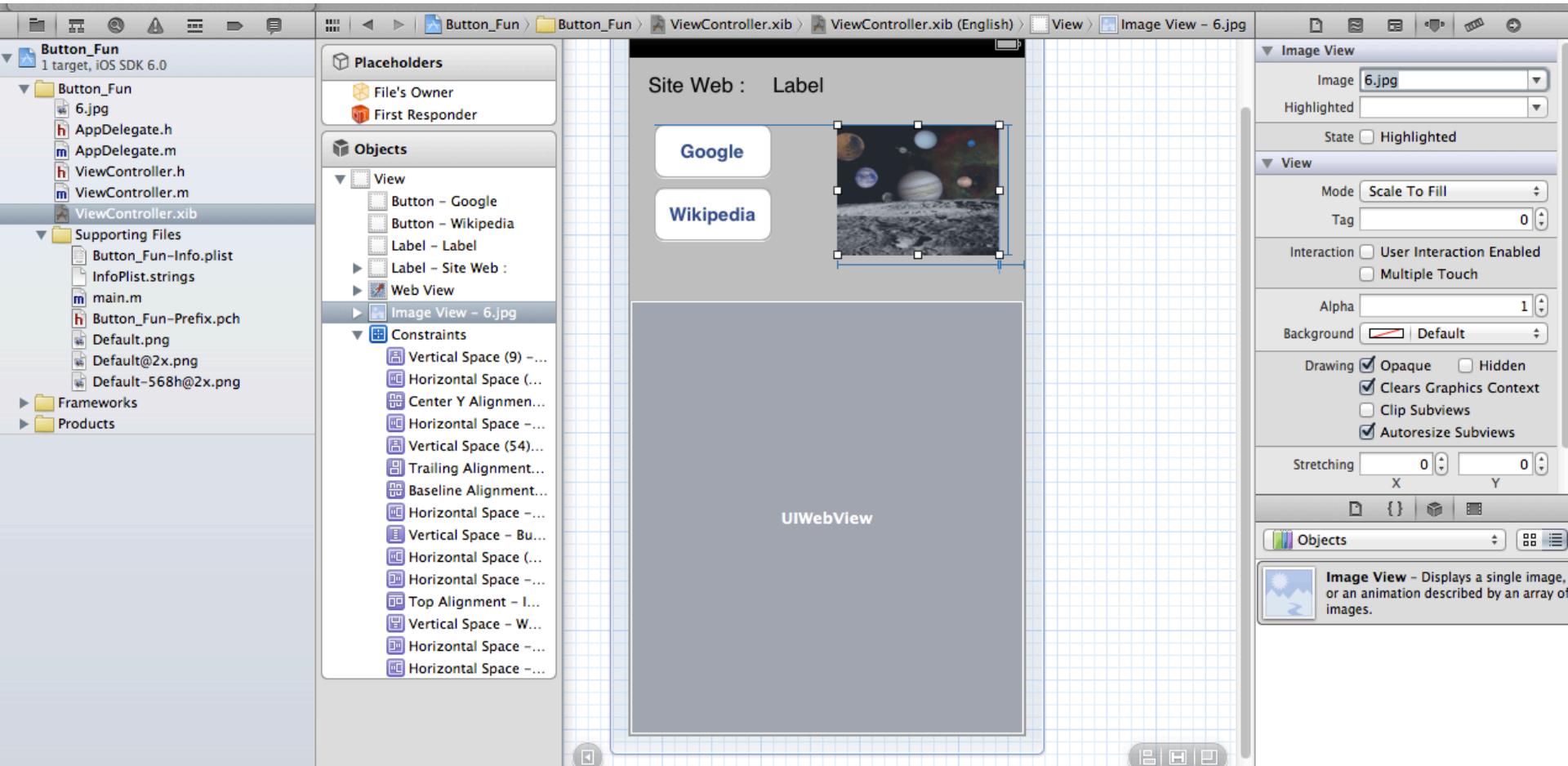
Utilisation d'application déléguées



Une application interactive !

Modification de l'application Button_Fun

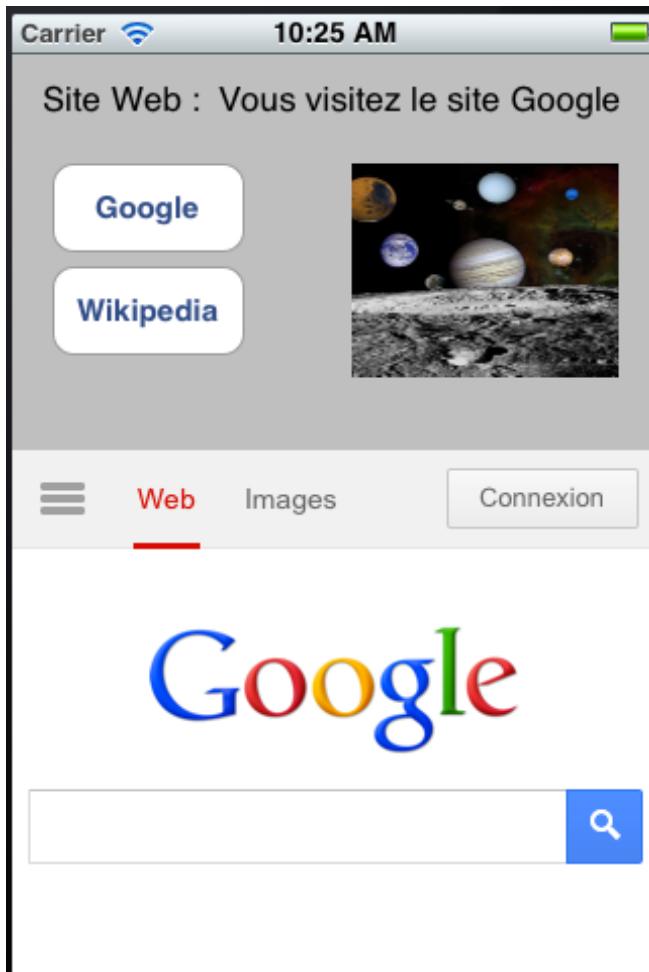
Présentation de l'interface



Une application interactive !

Modification de l'application Button_Fun

Présentation de l'interface



Exemple d'une requête pour l'affichage d'une vue web

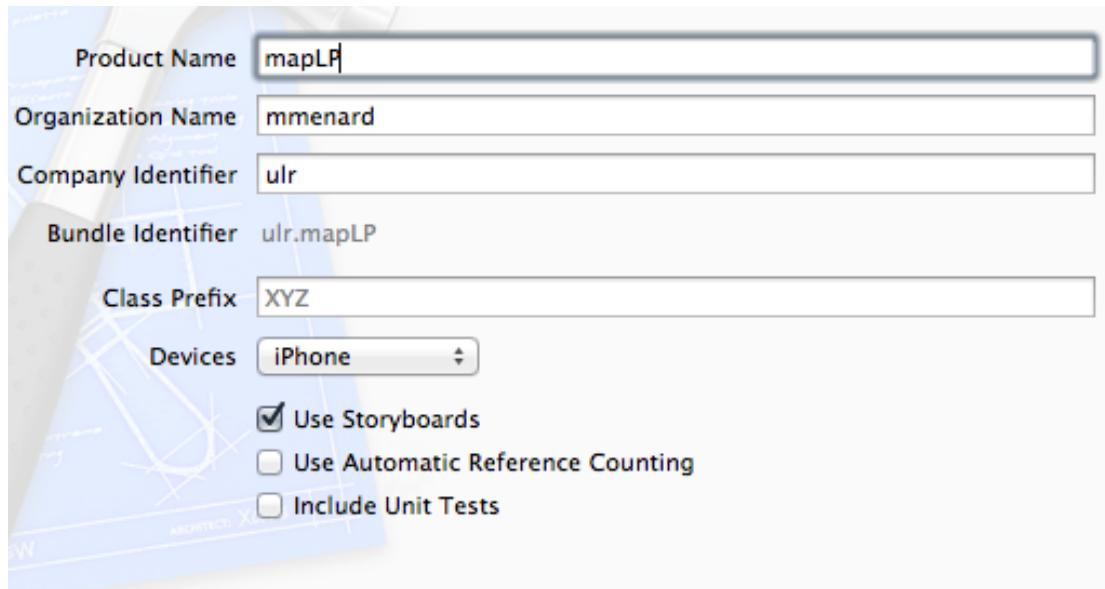
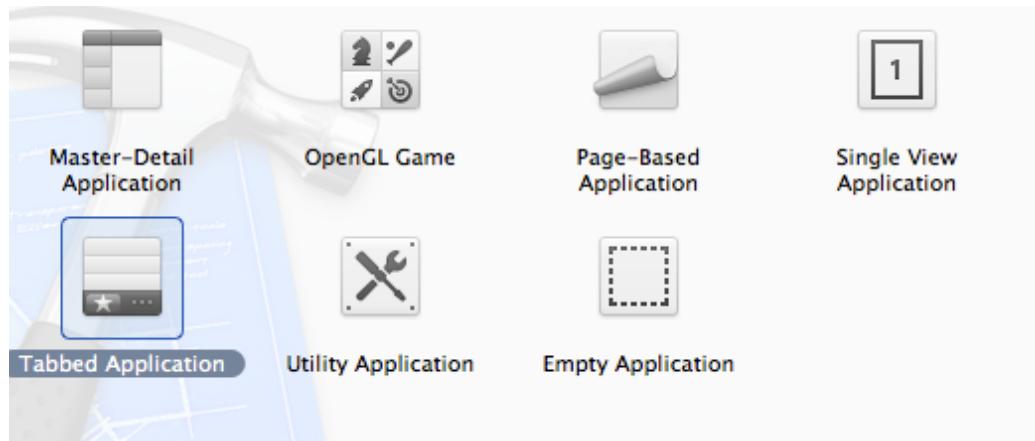
```
[webView loadRequest:[NSURLRequest  
requestWithURL:[NSURL  
URLWithString:@"http://www.google.fr"]]]];
```

Remarque :

*Pour comparer deux chaînes utilisez la méthode
isEqualTo*

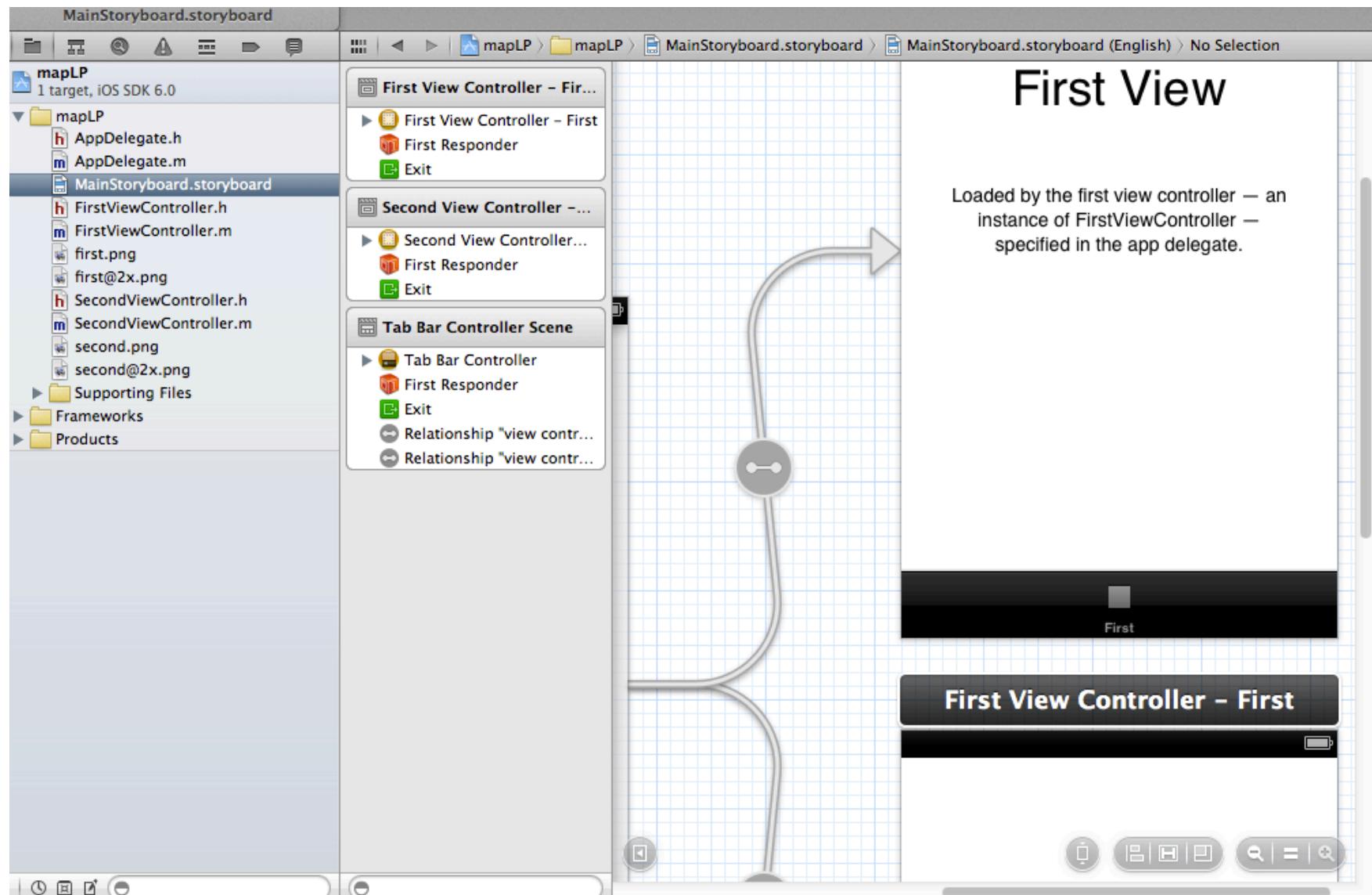
Une application interactive !

Une nouvelle application utilisant le storyBoard



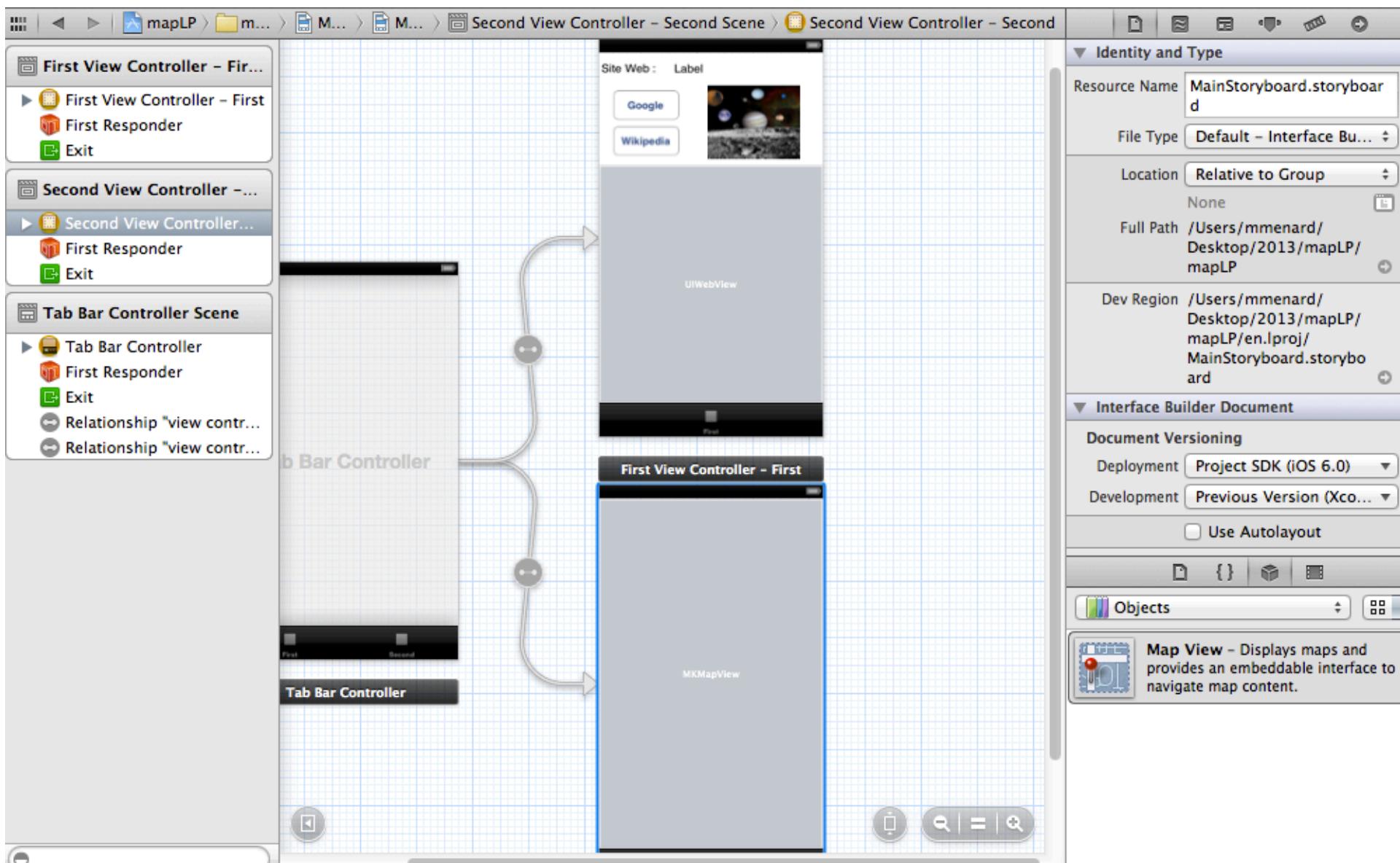
Une application interactive !

Une nouvelle application utilisant le storyBoard



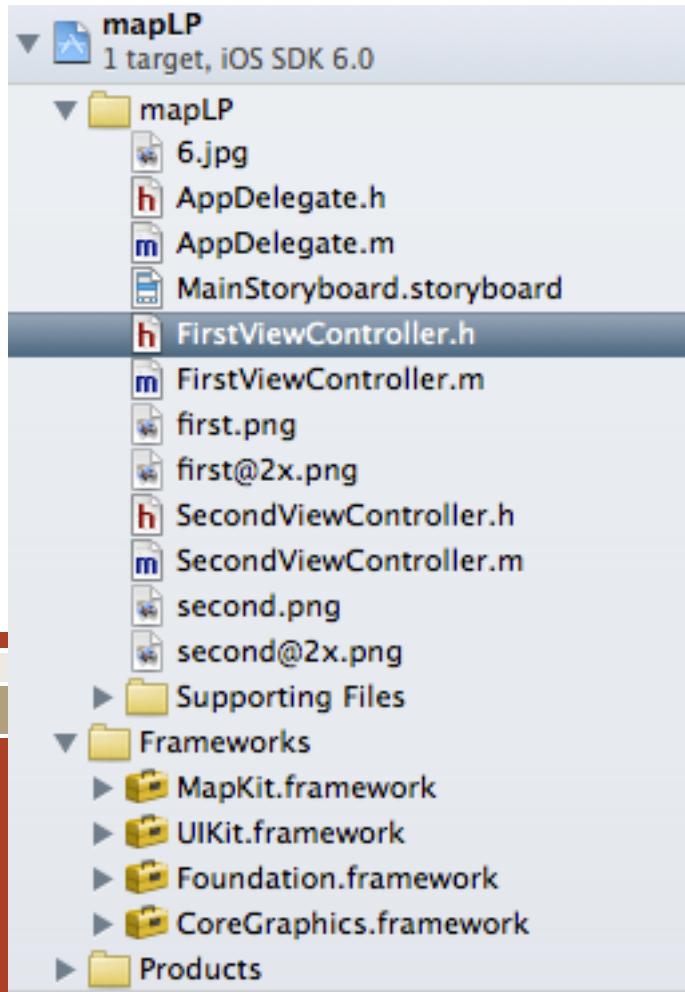
Une application interactive !

Une nouvelle application utilisant le storyBoard



Une application interactive !

Une nouvelle application utilisant le storyBoard



```
#import <UIKit/UIKit.h>

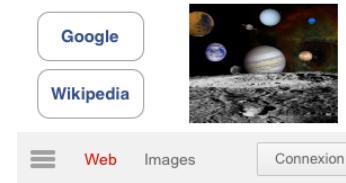
@interface FirstViewController : UIViewController{
    IBOutlet UILabel *statusText;
    IBOutlet UIWebView *webView;
}

@property(nonatomic,retain) UILabel *statusText;
@property(nonatomic,retain) IBOutlet UIWebView *webView;

-(IBAction)buttonPressedForWebView:(id)sender;

@end
```

Site Web : Vous visitez le site Google



Une application interactive !

Une nouvelle application utilisant le storyBoard

Ajouter le framework MapKit

The screenshot shows the Xcode interface. On the left, the Project Navigator displays the project 'mapLP' with its targets and files. A red arrow points from the 'Frameworks' section of the Project Navigator to the 'Link Binary With Libraries' section of the Build Phases tab on the right. The Build Phases tab shows the following configuration:

Frameworks	Dependencies
MapKit.framework	UIKit.framework
UIKit.framework	Foundation.framework
Foundation.framework	CoreGraphics.framework
CoreGraphics.framework	

Below the frameworks, there is a section for 'Add items' and 'Delete Resources'.

Code Snippet:

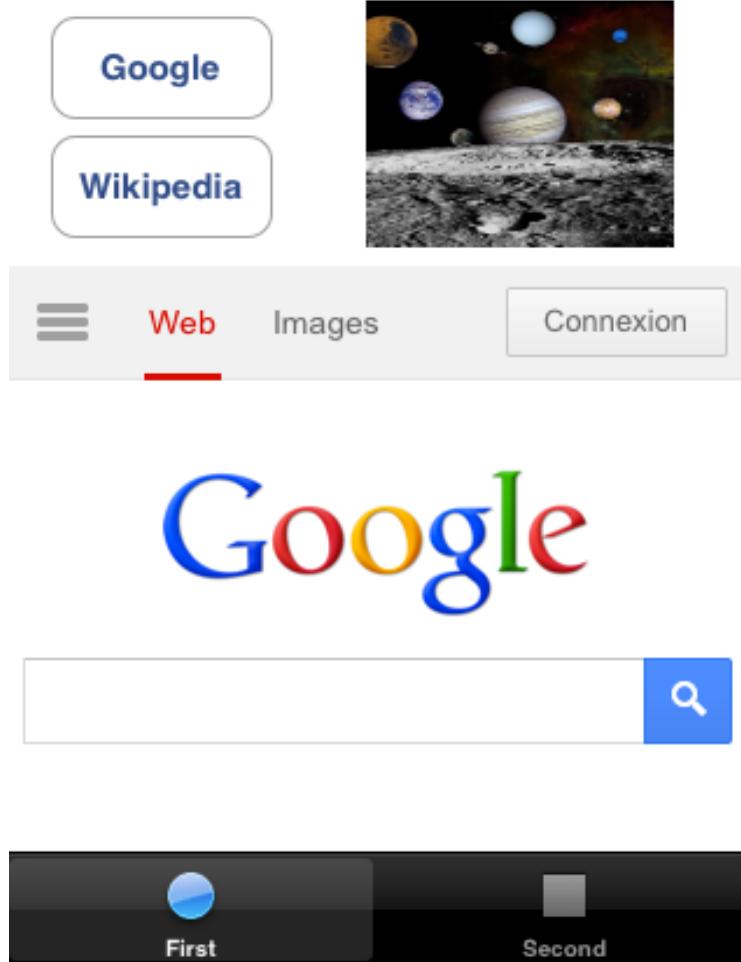
```
#import <UIKit/UIKit.h>
#import <MapKit/MapKit.h>

@interface SecondViewController : UIViewController
IBOutlet MKMapView *mapView;
@end
```

Une application interactive !

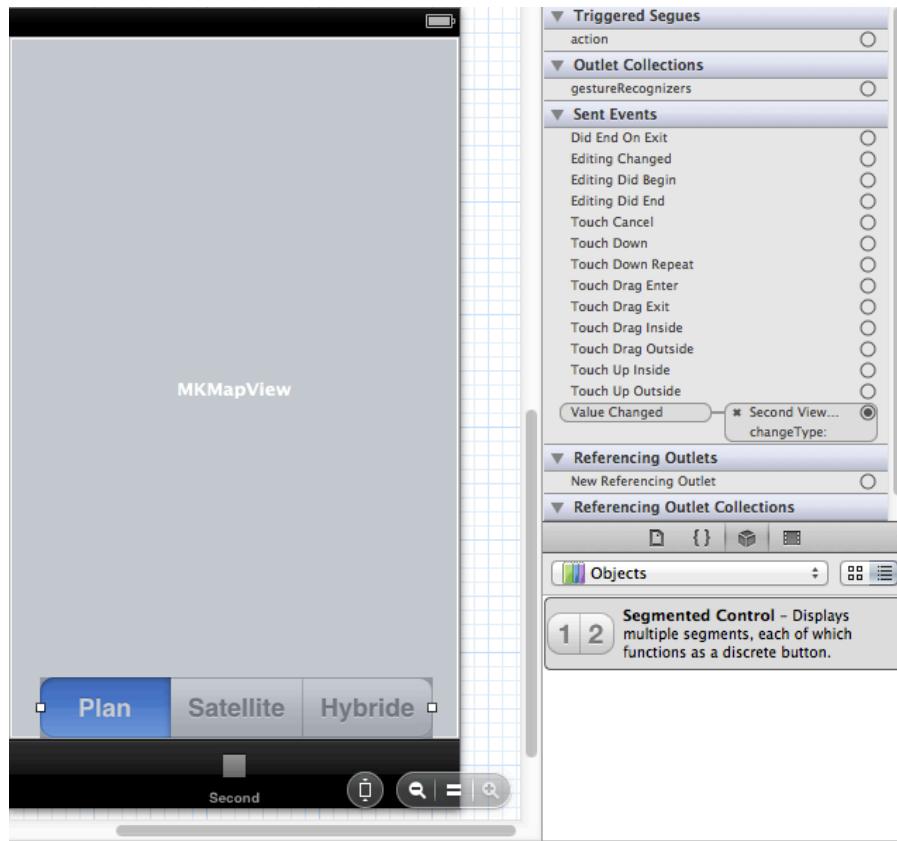
Une nouvelle application utilisant le storyBoard

Site Web : Vous visitez le site Google



Une application interactive !

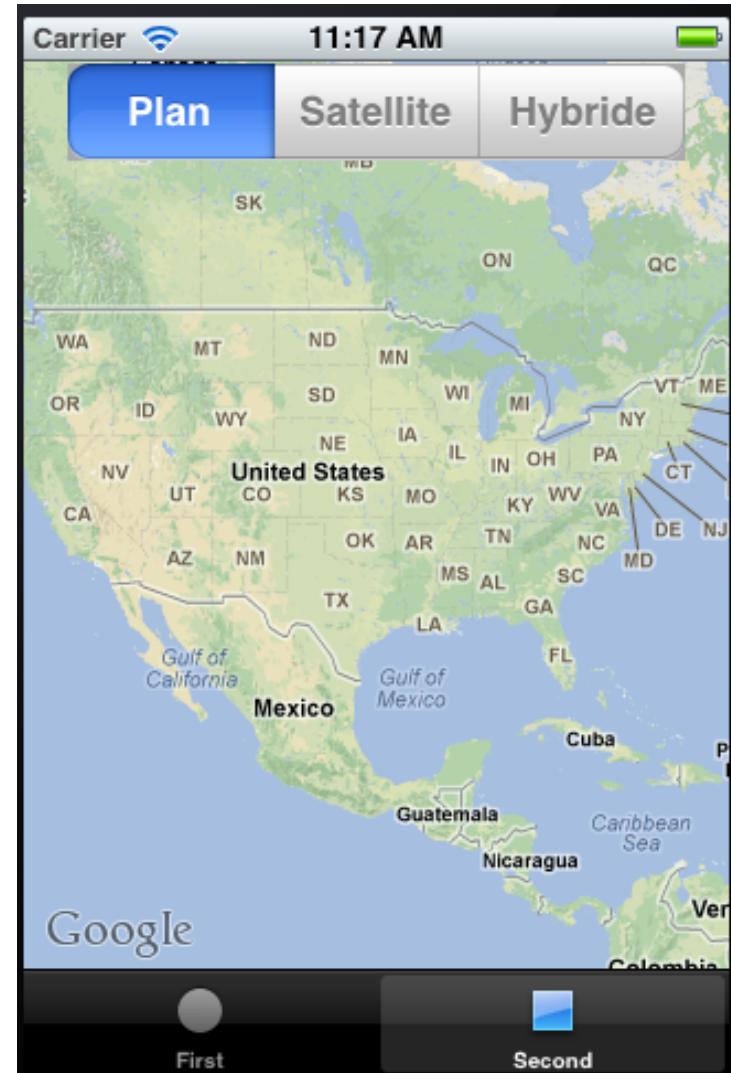
Une nouvelle application utilisant le storyBoard



```
- (IBAction) changeType:(id)sender{
    int val = [(UISegmentedControl *)sender selectedSegmentIndex];
    switch (val) {
        case 0:
            mapView.mapType=MKMapTypeStandard;
            break;
        case 1:
            mapView.mapType=MKMapTypeSatellite;
            break;
        case 2:
            mapView.mapType=MKMapTypeHybrid;
            break;
    }
}
```

Une application interactive !

Une nouvelle application utilisant le storyBoard



Une application interactive !

Une nouvelle application utilisant le storyBoard

```
@interface SecondViewController :  
UIViewController <MKMapViewDelegate>{
```

```
- (void)viewDidLoad  
{  
    [super viewDidLoad];  
    mapView.delegate=self;
```



```
- (void)mapView:(MKMapView *)mapView  
regionDidChangeAnimated:(BOOL)animated{  
  
    MKCoordinateRegion region =  
    mapView.region;  
  
    latitudeLabel.text = [NSString  
    stringWithFormat:@"%g",region.center.latitude];  
  
    longitudeLabel.text = [NSString  
    stringWithFormat:@"%g",region.center.longitude];  
  
    hauteurLabel.text = [NSString  
    stringWithFormat:@"%g",region.span.latitudeDelta];  
  
    largeurLabel.text = [NSString  
    stringWithFormat:@"%g",region.span.longitudeDelta];  
}
```

Une application interactive !

(Un écran et des contrôles...)

Une vue Image



Deux champs texte différents



Un slider



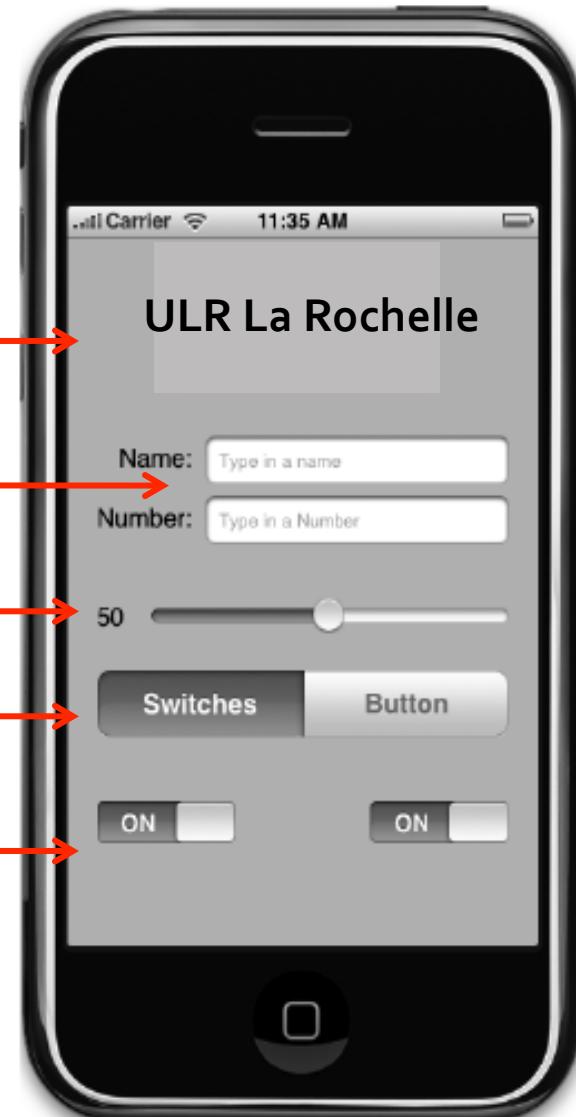
Un contrôle segmenté



Un couple de switch



Un bouton iphone



Une application interactive !

(Un écran et des contrôles...)

Une vue Image

Image statique

Deux champs texte différents

Texte alphanumérique

nombre

Un slider

Le label donne la valeur du slider

Un contrôle segmenté

Gestion de deux types de contrôles différents situés dans une zone

Un couple de switch

L'un change l'autre

Un bouton iphone



Une application interactive !

(Un écran et des contrôles...)



Action sheet !



Alerte !

Une application interactive !

(Un écran et des contrôles...)

Fenêtre hiérarchique pour grouper plusieurs items

Initialiser et rechercher les valeurs des contrôles (outlet, et argument sender des actions)

Contrôles d'états

Images déformables pour boutons

Cycle code-compile-debug

...

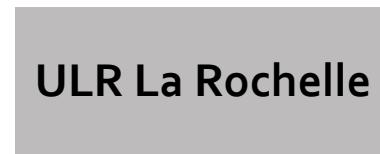


Une application interactive !

(Un écran et des contrôles...)



Créez une image .png
(<100,<300)



Additionnez l'image
au dossier ressource
(ou Add à partir du menu Project)

Chargez l'image via Interface Builder



Une application interactive !

Détermination des outlets

Les outlets doivent être définis dans le fichier d'entête du contrôleur avant de les connecter dans Interface Builder.

ULR La Rochelle

Name:

Number:

Statiques: ne doivent pas changer au cours de l'exécution : Outlets inutiles

Type in a name

Type in a Number

Pour accéder aux données que contiennent les labels, nous avons besoin d'outlets.



Une application interactive !

Control_FunViewController.h

```
#import <UIKit/UIKit.h>

@interface Control_FunViewController : UIViewController {
    UITextField      *nameField;
    UITextField      *numberField;
}

@property (nonatomic, retain) IBOutlet UITextField *nameField;
@property (nonatomic, retain) IBOutlet UITextField *numberField;
@end
```



Une application interactive !

Control_FunViewController.m

```
#import "Control_FunViewController.h"

@implementation Control_FunViewController
@synthesize nameField;
@synthesize numberField;
...
```

Déclarer avec le mot
retain dans les propriétés

```
- (void)dealloc {
    [nameField release];
    [numberField release];
    [super dealloc];
}
```



Une application interactive !

Détermination des actions

L'image et les deux labels n'interagissent pas
→ Pas d'actions

Les deux champs texte :

Pas de contrôle en dehors de l'affectation
d'un clavier adapté : numérique ou complet.
→ Pas d'actions

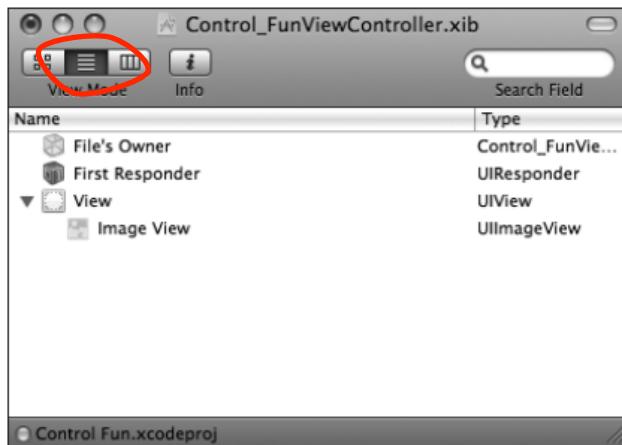


Une application interactive !

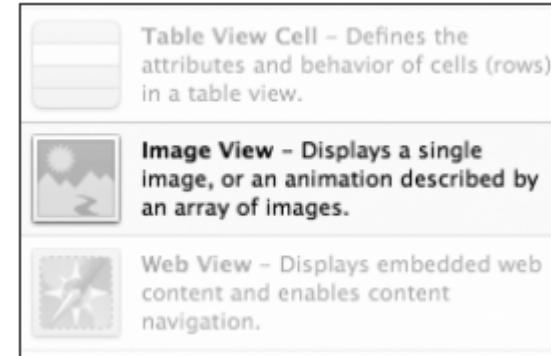
Construction de l'interface

Control_FunViewController.xib

Pour sélectionner l'image dans la vue, utilisez la vue hiérarchique du nib.



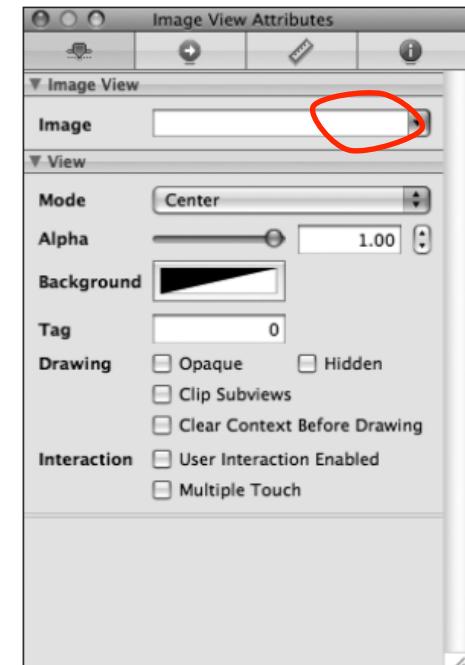
Library du menu Tool



Inspecteur:
options
éditable
de la classe
UIImageView

Resize : ⌘=
ou Size to Fit
du menu Layout

Alignment :
Align Horizontal Center in Container



Une application interactive !

Construction de l'interface

Control_FunViewController.xib

Autres options de positionnement

Show Bounds Rectangle

Touche option

Mode Attribute : le menu Mode (Center par défaut)

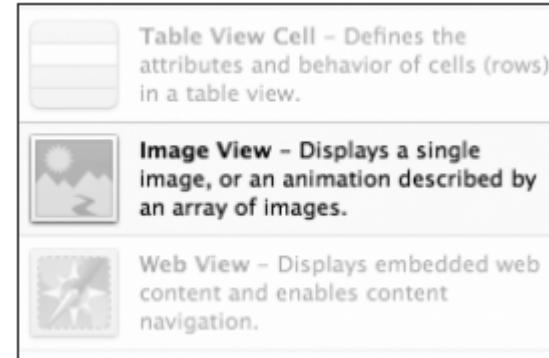
The alpha slider : *définition de la transparence*

Utiliser la valeur par défaut : 1

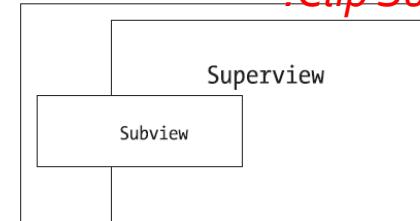
The tag Attribute: *permet d'identifier de façon unique un contrôle ou une vue (ex : quel contrôle est passé dans l'argument sender dans une méthode d'action ?).*

The tag Checkboxes: Opaque / optimisation, *!Hidden Checkbox, !Clip Subview, Autoresize, !User Interaction Enabled, !Multiple Touch*

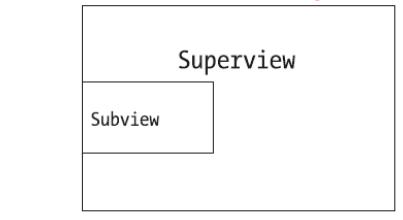
Library du menu Tool



!Clip Subview



Clip Subview



Une application interactive !

Construction de l'interface

Control_FunViewController.xib

*2 Text fields
2 labels*

Pour aligner la partie droite des deux labels

- Cliquez sur le label Name
- Appuyez sur la touche shift
- Cliquez sur le label Number
- Sous menu Alignement de Layout :
Align Right Edges



Une application interactive !

Construction de l'interface

Control_FunViewController.xib

Text fields

Texte par défaut

Texte affiché quand le champ n'a pas de valeur
(permet de clarifier l'action :
« Type in a name »/ « Type in a number »)

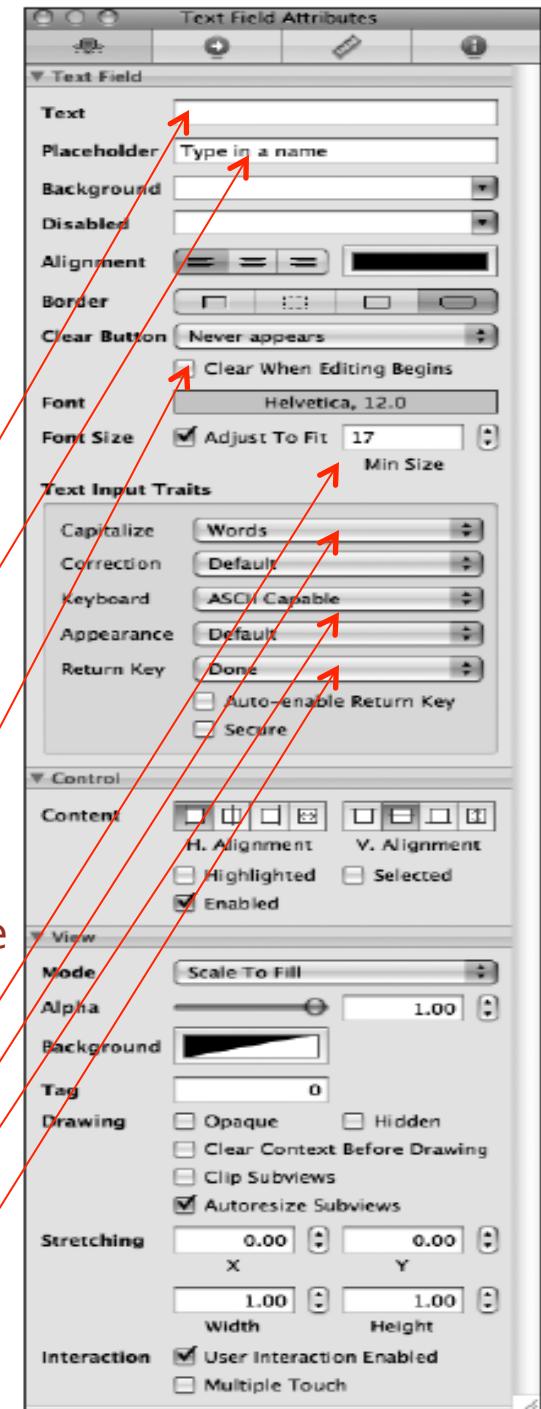
Spécifie ce qui se passe quand l'utilisateur touche ce
champ (ne pas sélectionnez ici)

Spécifie une taille minimum

Changer pour Words / valeur par défaut

Changer pour ASCII Capable / Number Pad

Changer pour Done / valeur par défaut



Une application interactive !

Connexion des Outlets

Control_FunViewController.xib

Créez les liens entre File's Owner vers chacun des champs texte

Connectez les vers les outlets correspondant

Sauvez et revenez à Xcode

Build and Run !!!!

Problème : le clavier est permanent !!!



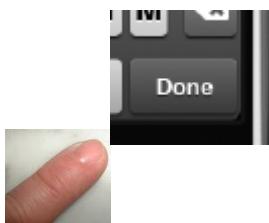
Une application interactive !

Gestion du clavier

Control_FunViewController.h

```
#import <UIKit/UIKit.h>

@interface Control_FunViewController : UIViewController {
    UITextField      *nameField;
    UITextField      *numberField;
}
@property (nonatomic, retain) IBOutlet UITextField *nameField;
@property (nonatomic, retain) IBOutlet UITextField *numberField;
- (IBAction)textFieldDoneEditing:(id)sender;
@end
```



Génération de l'évènement
Did End On Exit



Abandon du contrôle

Control_FunViewController.m

```
- (IBAction)textFieldDoneEditing:(id)sender{
    [sender resignFirstResponder];
}
```

Concept first responder : c'est le contrôle avec qui l'utilisateur interagit.

Quand un champ texte reçoit un status de first responder, le clavier associé à ce champ s'efface.

Une application interactive !

Gestion du clavier

Interface Builder



- Cliquez sur le champ texte Name et ⌘2
- Choisissez l'évènement **Did End On Exit** (au moment où l'utilisateur clique sur Done)
- Reliez le cercle à l'icône File's Owner
- Connectez le à l'action **TextFieldDoneEditing**:
- Même chose avec l'autre champ texte

Problème pour le champ texte Number !!!



Changement de la classe de l'objet contrôleur de vue : **UIView → UIControl**

Une application interactive !

Gestion du clavier

Control_FunViewController.h

```
#import <UIKit/UIKit.h>

@interface Control_FunViewController : UIViewController {
    UITextField      *nameField;
    UITextField      *numberField;
}
@property (nonatomic, retain) IBOutlet UITextField *nameField;
@property (nonatomic, retain) IBOutlet UITextField *numberField;
- (IBAction)textFieldDoneEditing:(id)sender;
- (IBAction)backgroundTap:(id)sender;
@end
```

Control_FunViewController.m

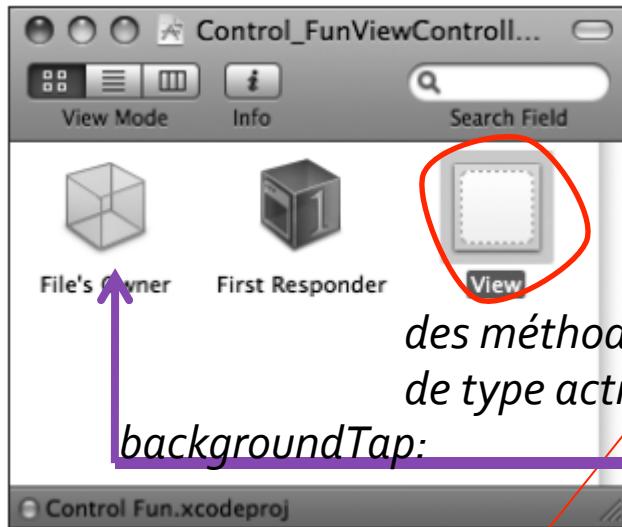
```
- (IBAction)backgroundTap:(id)sender {
    [nameField resignFirstResponder];
    [numberField resignFirstResponder];
}
```

Concept first responder : C'est le contrôle avec qui l'utilisateur interagit.

Quand un champ texte rapporte un status de first responder, le clavier associé à ce champ s'efface. L'action renvoie donc le statut first responder de l'un ou l'autre des champs texte.

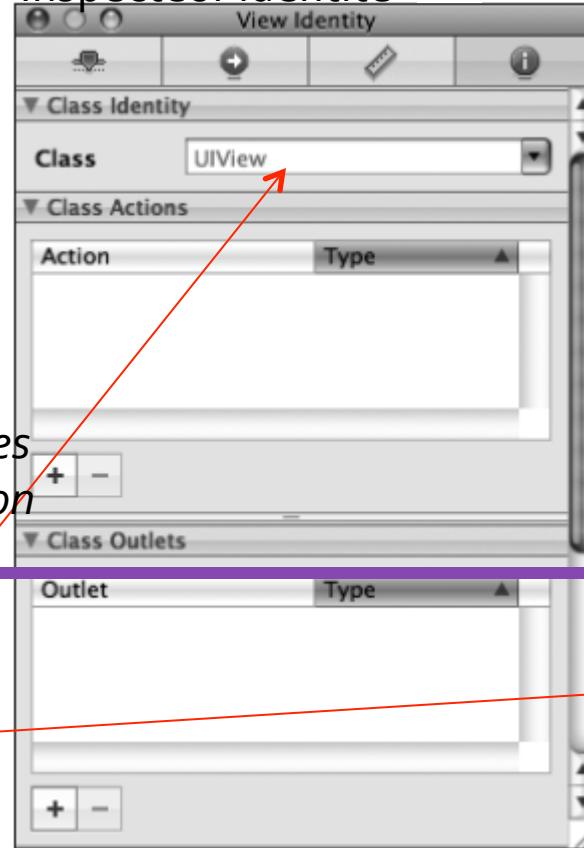
Une application interactive !

Gestion du clavier



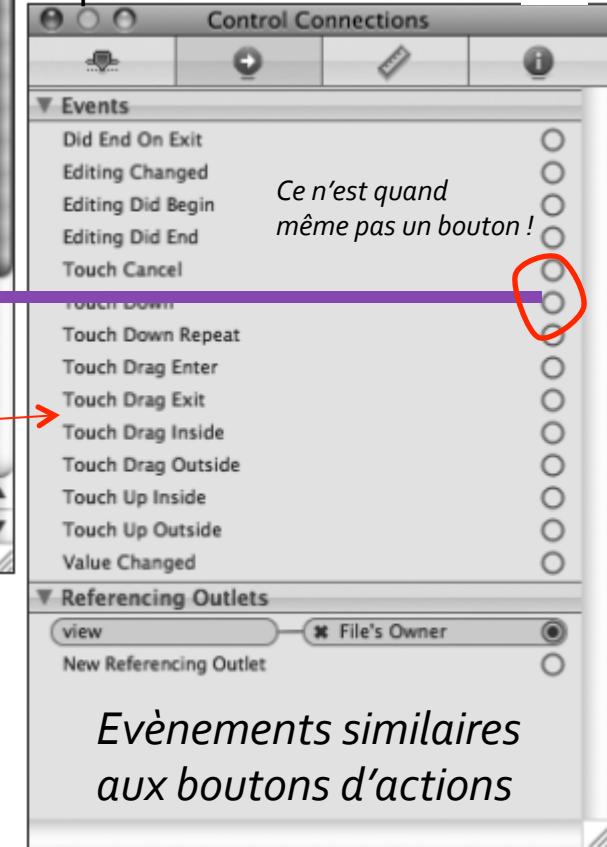
UIControl permet d'appeler des méthodes de type action à partir de tout événements standards

Inspecteur identité ⌘4



Permet de changer la classe sous-jacente de tout instance d'objet dans un nib.

Inspecteur de connexion ⌘2



Ce n'est quand même pas un bouton !

Evènements similaires aux boutons d'actions

Une application interactive !

Slider et Label

Le label a besoin d'être changé par programmation

→ Outlet

Le label est static : pas d'action

Le slider déclenche une action

→ l'action reçoit un pointeur sur le slider dans
l'argument sender

Question : avons-nous besoin d'accéder à la valeur
du slider à l'extérieur de la méthode d'action qu'il
appelle?



Une application interactive !

Slider et Label

Control_FunViewController.h

```
#import <UIKit/UIKit.h>

@interface Control_FunViewController : UIViewController {
    UITextField *nameField;
    UITextField *numberField;
    UILabel      *sliderLabel;
}
@property (nonatomic, retain) IBOutlet UITextField *nameField;
@property (nonatomic, retain) IBOutlet UITextField *numberField;
@property (nonatomic, retain) IBOutlet UILabel *sliderLabel;
- (IBAction)textFieldDoneEditing:(id)sender;
- (IBAction)backgroundTap:(id)sender;
- (IBAction)sliderChanged:(id)sender;
@end
```

Control_FunViewController.m

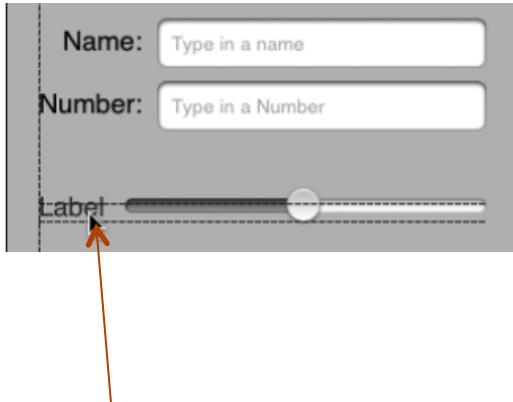
```
#import "Control_FunViewController.h"

@implementation Control_FunViewController
@synthesize nameField;
@synthesize numberField;
@synthesize sliderLabel;
- (IBAction)sliderChanged:(id)sender {
    UISlider *slider = (UISlider *)sender;
    int progressAsInt = (int)(slider.value + 0.5f);
    NSString *newText = [[NSString alloc] initWithFormat:@"%d",
    progressAsInt];
    sliderLabel.text = newText;
    [newText release];
}
- (IBAction)backgroundTap:(id)sender {
    ...
}
```

```
- (void)dealloc {
    [nameField release];
    [numberField release];
    [sliderLabel release];
    [super dealloc];
}
```

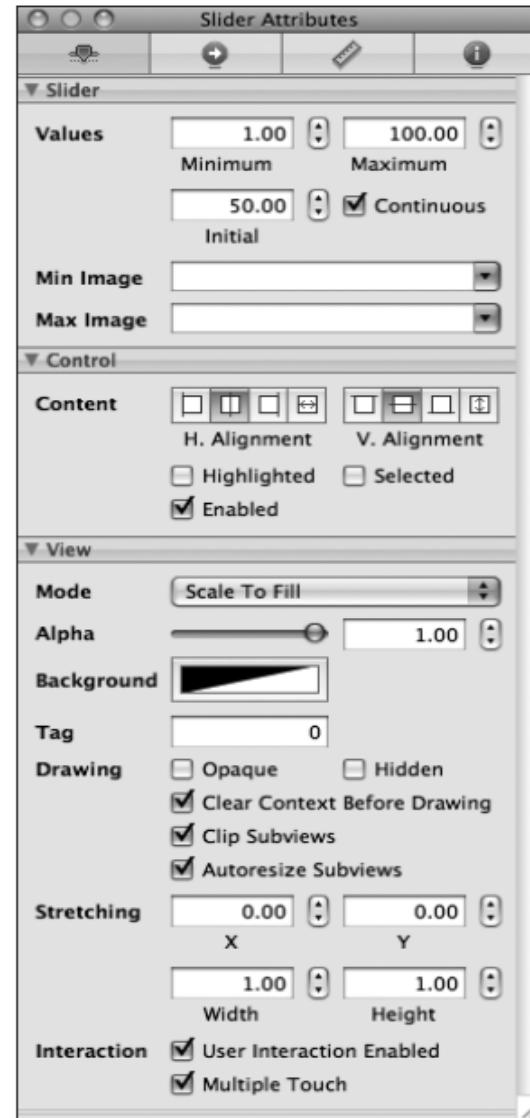
Une application interactive !

Slider et Label



100 puis 50 (vérification)

⌘=



Control-drag de l'icône
File'sOwner vers le label
Sélectionnez sliderLabel

Sélectionnez le slider
⌘2
Drag de Value Changed
vers File's Owner
Sélectionnez sliderChanged

Une application interactive !

Slider et Label

Slider entre 0 et 1.

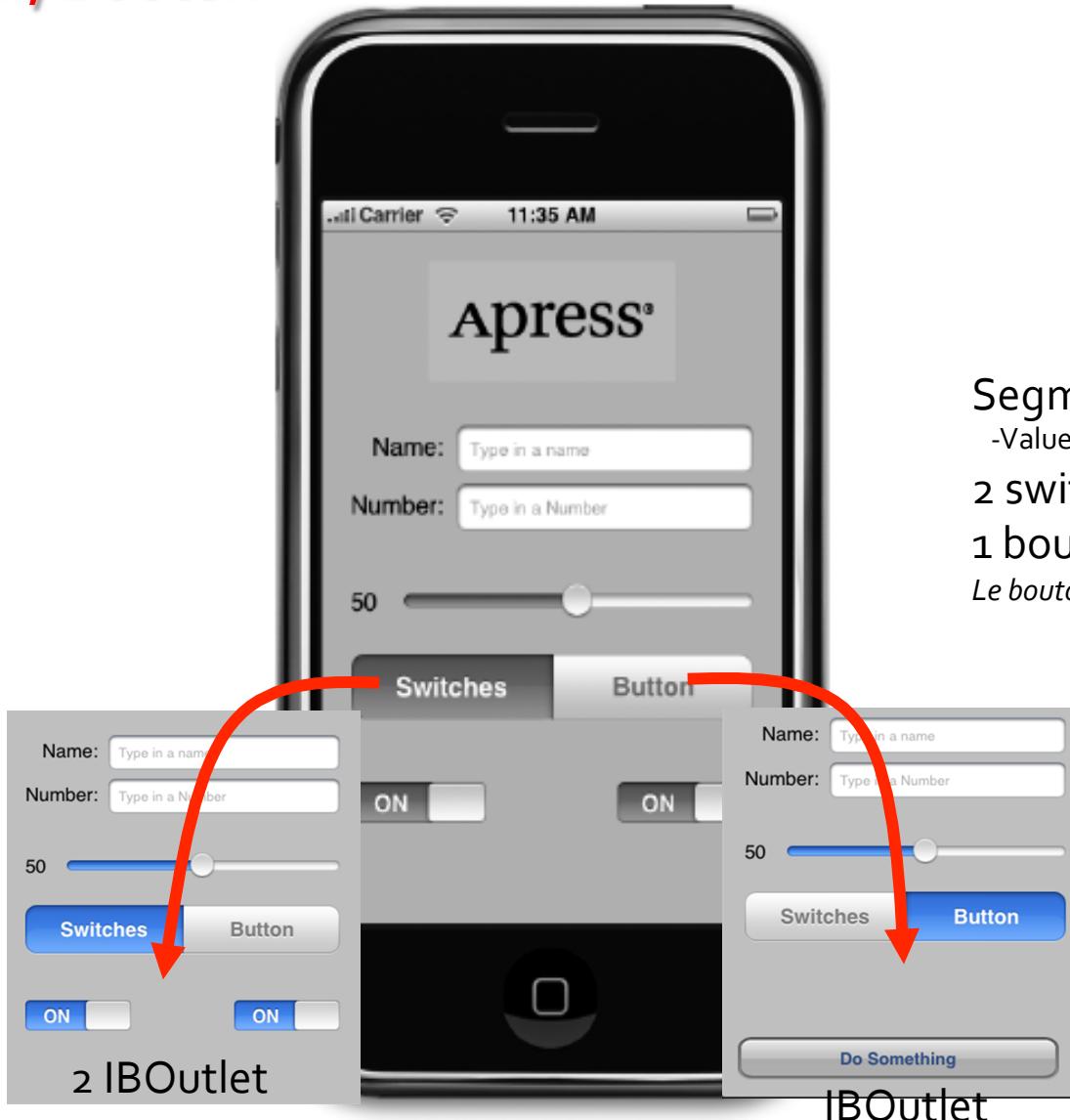
Autre méthode ...



```
-IBAction) sliderChanged:(id)sender {  
    float a=(float) [(UISlider *) sender value];  
  
    NSString *newText=[[NSString alloc] initWithFormat:@("%.2f",a];  
    sliderLabel.text=newText;  
    [newText release];  
}
```

Une application interactive !

Switch, Bouton et contrôle séquentiel



Segmented Control

-Value Changed → File's Owner

2 switches (Touch Up Inside)

1 bouton (Touch Up Inside)

Le bouton est caché au chargement de la vue

Segmented Control
-Value Changed → File's Owner
2 switchs (Touch Up Inside)
1 bouton (Touch Up Inside)

Une application interactive !

Switch, Bouton et contrôle segmenté

Control_FunViewController.h

```
#import <UIKit/UIKit.h>
#define kSwitchesSegmentIndex 0
@interface Control_FunViewController : UIViewController {
    UITextField      *nameField;
    UITextField      *numberField;
    UILabel         *sliderLabel;
    UISwitch        *leftSwitch;
    UISwitch        *rightSwitch;
    UIButton        *doSomethingButton;
}
@property (nonatomic, retain) IBOutlet UITextField *nameField;
@property (nonatomic, retain) IBOutlet UITextField *numberField;
@property (nonatomic, retain) IBOutlet UILabel *sliderLabel;
@property (nonatomic, retain) IBOutlet UISwitch *leftSwitch;
@property (nonatomic, retain) IBOutlet UISwitch *rightSwitch;
@property (nonatomic, retain) IBOutlet UIButton *doSomethingButton;
- (IBAction)textFieldDoneEditing:(id)sender;
- (IBAction)backgroundTap:(id)sender;
- (IBAction)sliderChanged:(id)sender;
- (IBAction)toggleControls:(id)sender;
- (IBAction)switchChanged:(id)sender;
- (IBAction)buttonPressed;
@end
```

Control_FunViewController.m

```
#import "Control_FunViewController.h"

@implementation Control_FunViewController
@synthesize nameField;
@synthesize numberField;
@synthesize sliderLabel;

- (void)dealloc {
    [nameField release];
    [numberField release];
    [sliderLabel release];
    [leftSwitch release];
    [rightSwitch release];
    [doSomethingButton release];
    [super dealloc];
}

-synthesize leftSwitch;
@synthesize rightSwitch;
@synthesize doSomethingButton;
- (IBAction)toggleControls:(id)sender {
    if ([sender selectedSegmentIndex] == kSwitchesSegmentIndex)
    {
        leftSwitch.hidden = NO;
        rightSwitch.hidden = NO;
        doSomethingButton.hidden = YES;
    }
    else
    {
        leftSwitch.hidden = YES;
        rightSwitch.hidden = YES;
        doSomethingButton.hidden = NO;
    }
}

- (IBAction)switchChanged:(id)sender {
    UISwitch *whichSwitch = (UISwitch *)sender;
    BOOL setting = whichSwitch.isOn;
    [leftSwitch setOn:setting animated:YES];
    [rightSwitch setOn:setting animated:YES];
}

- (IBAction)buttonPressed {
    // TODO: Implement Action Sheet and Alert
}

- (IBAction)sliderChanged:(id)sender {
    ...
}
```

Une application interactive !

Alerte!

The screenshot shows a code editor window for a file named `Control_FunViewController.m`. The code is written in Objective-C and defines a class `Control_FunViewController` with various synthesized properties and an `IBAction` method. A callout bubble is overlaid on the screen, containing a list of methods with the title "TODO: Implement Action Sheet and Alert".

```
// Control_FunViewController.m
// Control Fun
//
// Created by jeff on 3/30/09.
// Copyright Jeff LaMarche 2009. All rights reserved.

#import "Control_FunViewController.h"

@implementation Control_FunViewController
@synthesize nameField;
@synthesize numberField;
@synthesize sliderLabel;
@synthesize leftSwitch;
@synthesize rightSwitch;
@synthesize doSomethingButton;

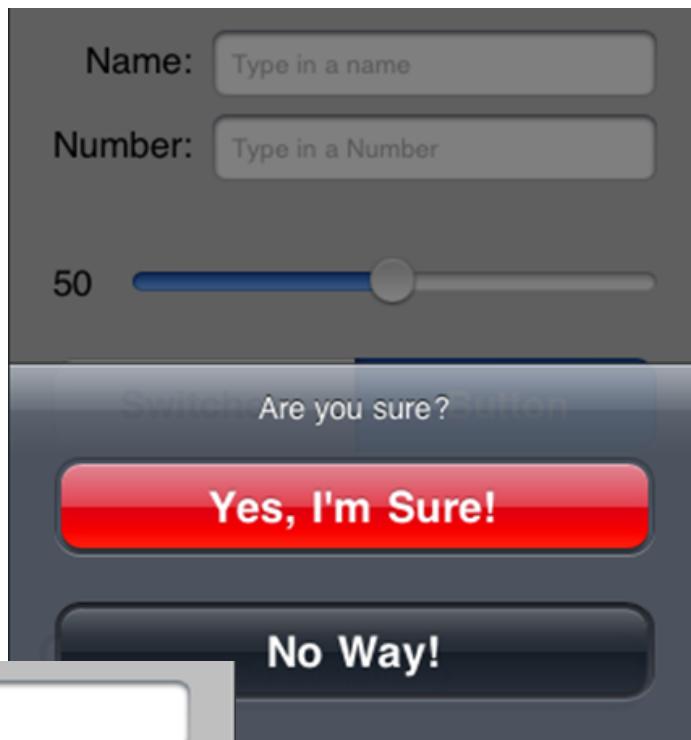
- (IBAction)toggleControls:(id)sender {

    if ([sender selectedSegmentIndex] == kSwitchesSegmentIndex)
    {
        leftSwitch.hidden = NO;
        rightSwitch.hidden = NO;
        doSomethingButton.hidden = YES;
    }
    else
    {
        leftSwitch.hidden = YES;
    }
}
```

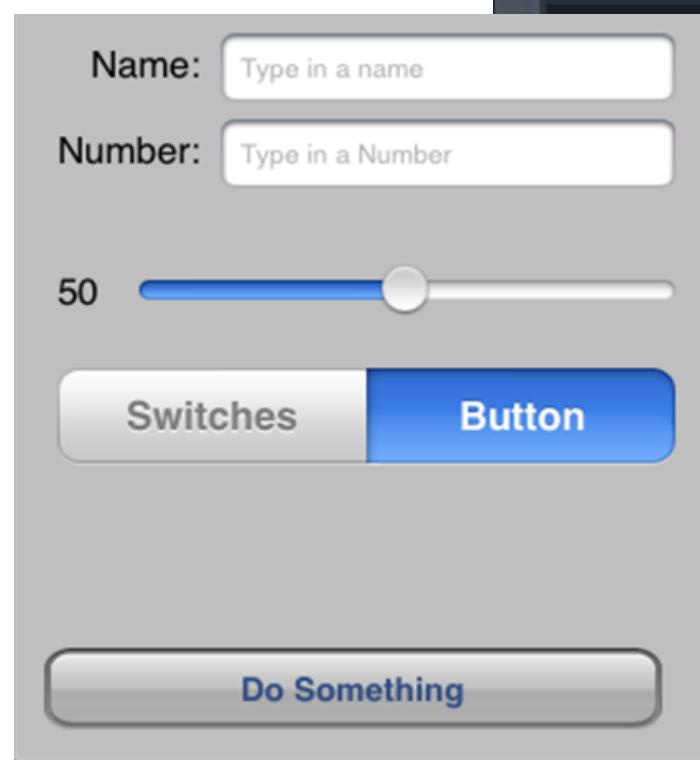
TODO: Implement Action Sheet and Alert

- toggleControls:
- switchChanged:
- buttonPressed
- sliderChanged:
- textFieldDoneEditing:
- backgroundClick:
- didReceiveMemoryWarning
- viewDidUnload
- dealloc

```
- (IBAction) buttonPressed {
// TODO: Implement Action Sheet and Alert
UIActionSheet *actionSheet = [[UIActionSheet alloc]
initWithTitle:@"Are you sure?"
delegate:self
cancelButtonTitle:@"No Way!"
destructiveButtonTitle:@"Yes, I'm Sure!"
otherButtonTitles:nil];
[actionSheet showInView:self.view];
[actionSheet release];
}
```



Vous devez implémenter le protocole UIActionSheetDelegate



*Ce que vous devez
(s)avoir avant de
commencer ...*

*Quelques mots
sur objective-C*

*Premiers projets:
Hello
Application
interactive*

Plus loin...

*Autorotation
Photo
Audio, vidéo*



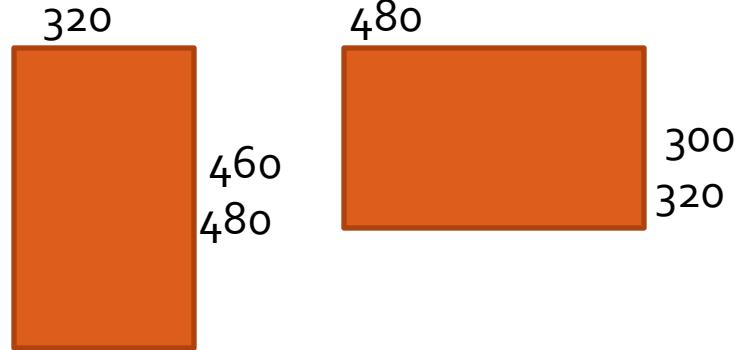
Autorotation et autoredimensionnement!

Portrait/paysage



Cette option doit être utile !!

Doit être spécifiée dans le contrôleur de vue.
La vue sera redimensionnée.



Vous devez gérer un minimum de contrôle de l'affichage. 3 approches possibles :

- spécifier l'attribut autosize
- repositionner les objets
- écrire deux versions différentes

Autorotation et autoredimensionnement!

Attribut resize

Notre vue supporte l'autorotation :

AutosizeViewController.m

```
/*  
// Override to allow orientations other than the default portrait  
// orientation.  
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation {  
    // Return YES for supported orientations  
    return (interfaceOrientation == UIInterfaceOrientationPortrait);  
}  
*/  
par defaut  
...
```

4 rotations sont définies :

- **UIInterfaceOrientationPortrait**
- **UIInterfaceOrientationPortraitUpsideDown**
- **UIInterfaceOrientationLandscapeLeft**
- **UIInterfaceOrientationLandscapeRight**



shouldAutorotateToInterfaceOrientation:

?

- **UIInterfaceOrientationPortrait**
- **UIInterfaceOrientationPortraitUpsideDown**
- **UIInterfaceOrientationLandscapeLeft**
- **UIInterfaceOrientationLandscapeRight**

yes or no
shouldAutorotateToInterfaceOrientation:

Autorotation et autoredimensionnement!

Attribut resize

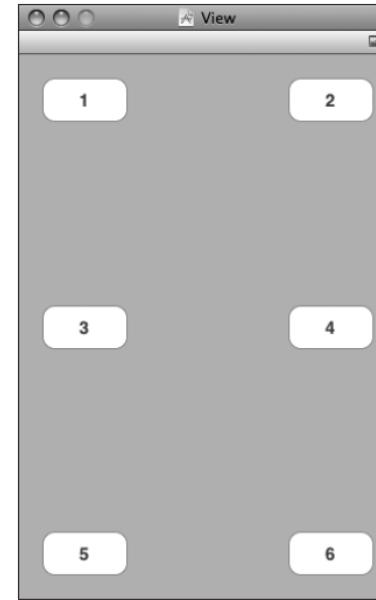
Notre vue supporte l'autorotation :

AutosizeViewController.m



```
- (BOOL)shouldAutorotateToInterfaceOrientation:  
    (UIInterfaceOrientation)interfaceOrientation {  
    return YES;  
}
```

Retourne YES pour toutes les rotations.

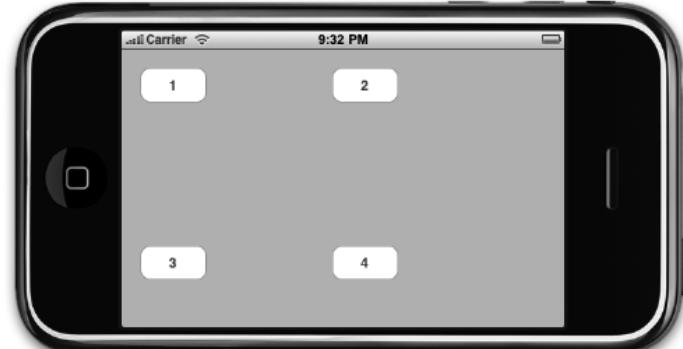


AutoSizeViewController.m
et Interfaceuilder

```
- (BOOL)shouldAutorotateToInterfaceOrientation:  
    (UIInterfaceOrientation)interfaceOrientation {  
    return (interfaceOrientation !=  
        UIInterfaceOrientationPortraitUpsideDown);  
}
```

Mode portrait et mode paysage
mais pas UpsideDown pour le mode portrait

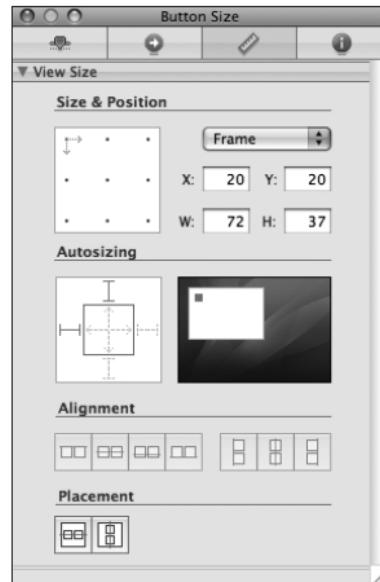
Sur le simulateur, lancer Rotate Left du menu Hardware



Autorotation et autoredimensionnement!

Attribut resize

#3

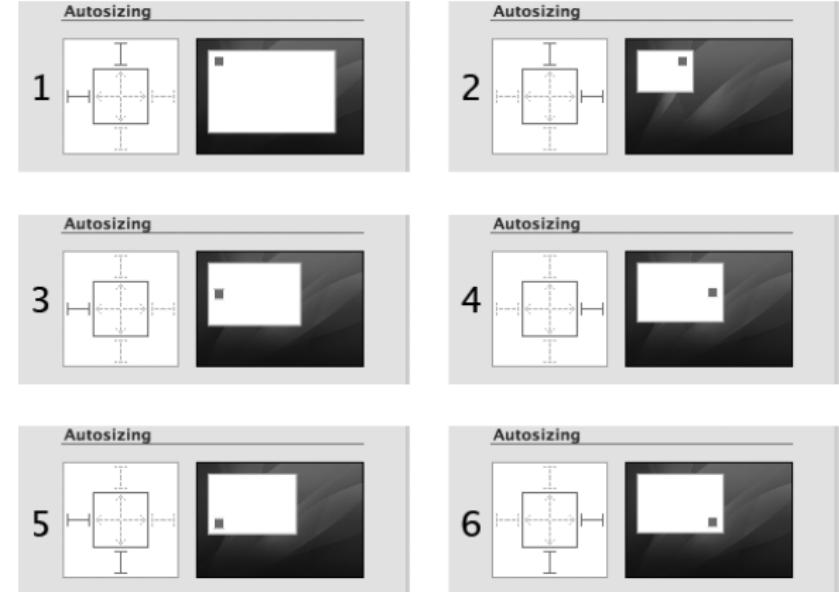


Par défaut

Test 1 + simulateur

Test 2

The first screenshot shows the default configuration where the button's width and height are both constrained by springs. The second screenshot shows a configuration where the width is constrained by a spring and the height is fixed (indicated by a dashed line). The third screenshot shows a configuration where the height is constrained by a spring and the width is fixed.

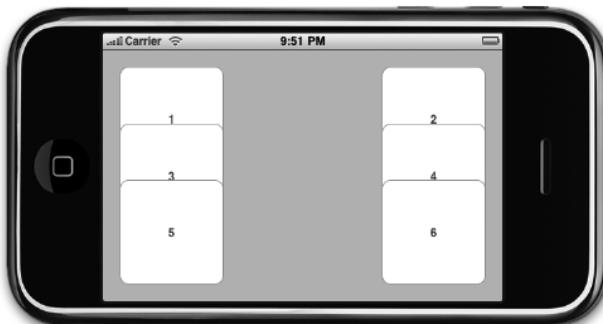
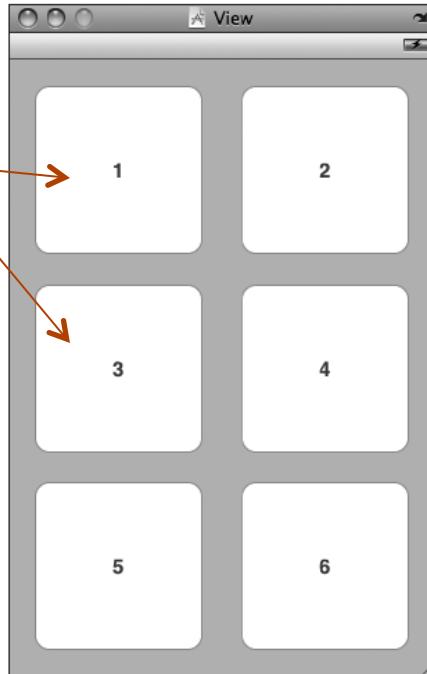


Autorotation et autoredimensionnement!

Attribut resize et restructuration de la vue

Interface Builder
size inspector

125*125
pour chaque bouton



Déclaration et connexion des outlets afin de changer un attribut de contrôle.

AutosizeViewController.m

```
#import <UIKit/UIKit.h>

@interface AutosizeViewController : UIViewController {
    UIButton *button1;
    UIButton *button2;
    UIButton *button3;
    UIButton *button4;
    UIButton *button5;
    UIButton *button6;
}
@property (nonatomic, retain) IBOutlet UIButton *button1;
@property (nonatomic, retain) IBOutlet UIButton *button2;
@property (nonatomic, retain) IBOutlet UIButton *button3;
@property (nonatomic, retain) IBOutlet UIButton *button4;
@property (nonatomic, retain) IBOutlet UIButton *button5;
@property (nonatomic, retain) IBOutlet UIButton *button6;
@end
```

Interface Builder : Control-drag de l'icône File's Owner vers chacun des six boutons.

Autorotation et autoredimensionnement!

Attribut resize et restructuration de la vue

AutosizeViewController.m

willAnimateRotationToInterfaceOrientation:duration:



Méthode appelée automatiquement après une rotation
Mais avant que l'animation finale ait lieu.

```
#import "AutosizeViewController.h"

@implementation AutosizeViewController
@synthesize button1;
@synthesize button2;
@synthesize button3;
@synthesize button4;
@synthesize button5;
@synthesize button6;
```

toInterfaceOrientation

```
- (void)willAnimateRotationToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation duration:(NSTimeInterval)duration {
    if (interfaceOrientation == UIInterfaceOrientationPortrait
        || interfaceOrientation ==
           UIInterfaceOrientationPortraitUpsideDown) {
        button1.frame = CGRectMake(20, 20, 125, 125);
        button2.frame = CGRectMake(175, 20, 125, 125);
        button3.frame = CGRectMake(20, 168, 125, 125);
        button4.frame = CGRectMake(175, 168, 125, 125);
        button5.frame = CGRectMake(20, 315, 125, 125);
        button6.frame = CGRectMake(175, 315, 125, 125);
    }
    else {
        button1.frame = CGRectMake(20, 20, 125, 125);
        button2.frame = CGRectMake(20, 155, 125, 125);
        button3.frame = CGRectMake(177, 20, 125, 125);
        button4.frame = CGRectMake(177, 155, 125, 125);
        button5.frame = CGRectMake(328, 20, 125, 125);
        button6.frame = CGRectMake(328, 155, 125, 125);
    }
}
```

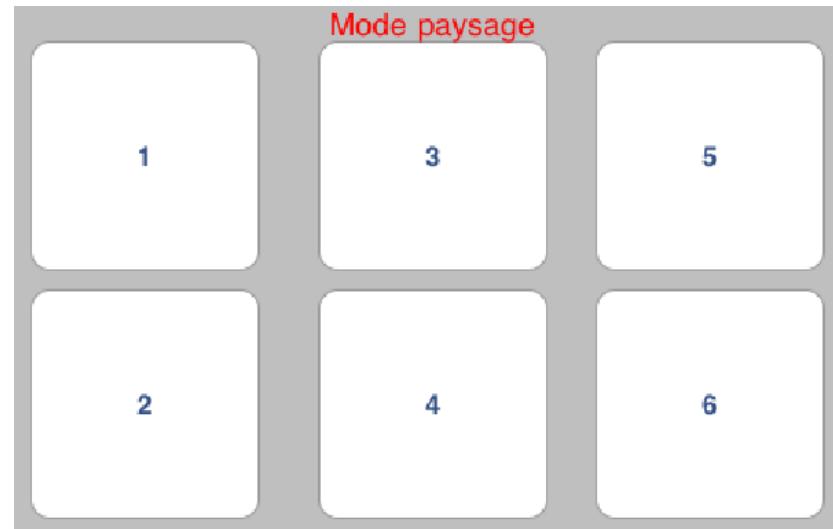
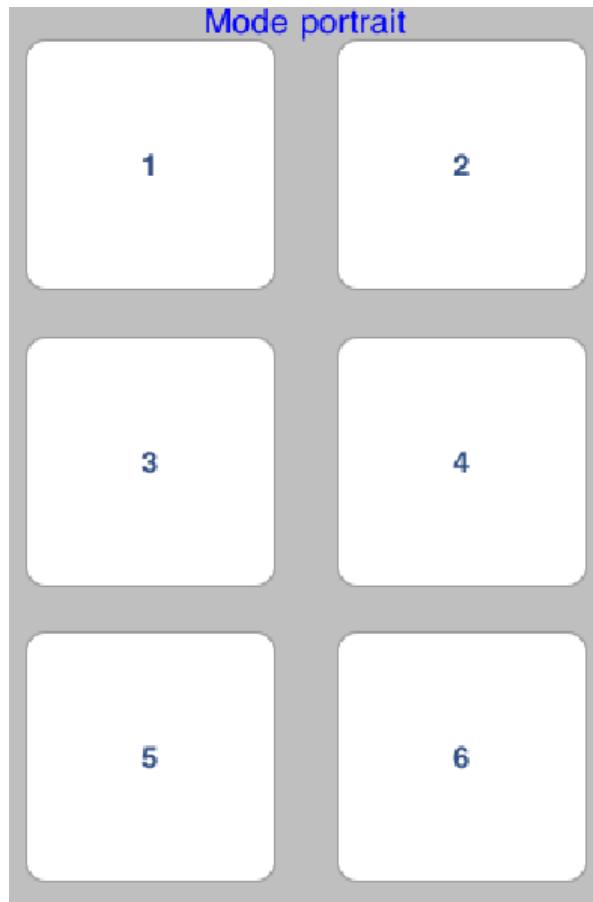
Propriété
frame
(CGRect
structure)

```
- (void)dealloc {
    [button1 release];
    [button2 release];
    [button3 release];
    [button4 release];
    [button5 release];
    [button6 release];
    [super dealloc];
}
```

Autorotation et autoredimensionnement!

Attribut `resize` et restructuration de la vue

Nouvelle version : modifiez l'application



Autorotation et autoredimensionnement!

Swapping de vues

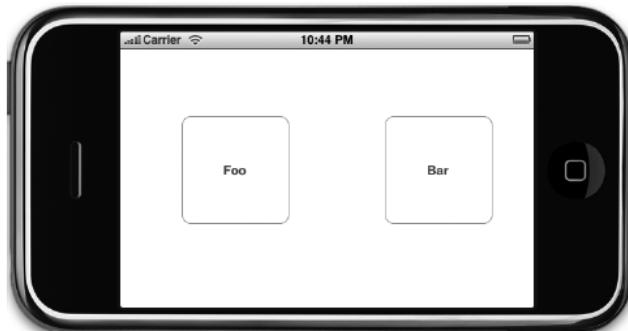
Les deux vues sont différentes !!

Nécessite ici :

- deux vues
- deux boutons par vue
- deux outlets par bouton

Lorsque j'appuie sur un bouton et que je tourne l'Iphone,

.....



Déclaration des actions et des outlets
(voir le code complet en annexe)

```
#import <UIKit/UIKit.h>
#define degreesToRadians(x) (M_PI * (x) / 180.0)
@interface SwapViewController : UIViewController {
    UIView      *landscape;
    UIView      *portrait;
    // Foo
    UIButton   *landscapeFooButton;
    UIButton   *portraitFooButton;
    // Bar
    UIButton   *landscapeBarButton;
    UIButton   *portraitBarButton;
}
@property (nonatomic, retain) IBOutlet UIView *landscape;
@property (nonatomic, retain) IBOutlet UIView *portrait;
@property (nonatomic, retain) IBOutlet UIButton *landscapeFooButton;
@property (nonatomic, retain) IBOutlet UIButton *portraitFooButton;
@property (nonatomic, retain) IBOutlet UIButton *landscapeBarButton;
@property (nonatomic, retain) IBOutlet UIButton *portraitBarButton;
-(IBAction)buttonPressed:(id)sender;
@end

#define degreesToRadians(x) (M_PI * (x) / 180.0)
```

Autorotation et autoredimensionnement!

Swapping de vues

```
- (void)willAnimateRotationToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation duration:(NSTimeInterval)duration {
    if (interfaceOrientation == UIInterfaceOrientationPortrait)
    {
        self.view = self.portrait;

        self.view.transform = CGAffineTransformMakeRotation(degreesToRadian(0));
        self.view.bounds = CGRectMake(0.0, 0.0, 300.0, 480.0);
    }
    else if (interfaceOrientation == UIInterfaceOrientationLandscapeLeft)
    {
        self.view = self.land;
        self.view.transform = CGAffineTransformMakeRotation(degreesToRadian(-90));
        self.view.bounds = CGRectMake(0.0, 0.0, 460.0, 320.0);
    }
    else if (interfaceOrientation == UIInterfaceOrientationPortraitUpsideDown)
    {
        self.view = self.portrait;

        self.view.transform = CGAffineTransformMakeRotation(degreesToRadian(180));
        self.view.bounds = CGRectMake(0.0, 0.0, 300.0, 480.0);
    }
    else if (interfaceOrientation == UIInterfaceOrientationLandscapeRight)
    {
        self.view = self.land;
        self.view.transform =
CGAffineTransformMakeRotation(degreesToRadian(90));
        self.view.bounds = CGRectMake(0.0, 0.0, 460.0, 320.0);
    }
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation {
    return YES;
}
```

toInterfaceOrientation

Appareil photo et librairie

Image picker

Sélectionner une image à partir d'une source

Classe UIImagePickerController :

- créer une instance de la classe
- spécifier son image source
- prise de contrôle de l'iphone par Image Picker
- édition basique : scaling, cropping
- envoie au délégué
- le délégué a la responsabilité de déconnecter Image Picker
- retour à l'application



Deux sources d'images :
liste d'images et appareil photo.

Appareil photo et librairie

Création de UIImagePickerController

Vérifier si l'appareil supporte la source

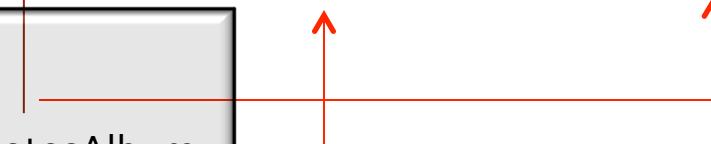
Librairie photo disponible ?

```
if ([UIImagePickerController isSourceTypeAvailable:  
    UIImagePickerControllerSourceTypePhotoLibrary]) {
```

La méthode isSourceTypeAvailable retourne
Yes or No

Deux autres valeurs possibles :

- UIImagePickerControllerSourceTypeCamera
- UIImagePickerControllerSourceTypeSavedPhotosAlbum
(les plus récentes)



Appareil photo et librairie

Création de UIImagePickerController

Lancement de l'image picker

```
UIImagePickerController *picker = [[UIImagePickerController alloc] init];
picker.delegate = self;
picker.sourceType = UIImagePickerControllerSourceTypePhotoLibrary;
[self presentModalViewController:picker animated:YES];
[picker release];
```

Présentation de l'image picker à l'utilisateur

presentModalViewController:animated: (hérite de UIView)

Appareil photo et librairie

Implémentation du délégué du contrôleur de Image Picker

Protocole UIImagePickerControllerDelegate

Deux méthodes

imagePickerController:didFinishPickingImage:editingInfo:

Est appelée quand l'utilisateur a pris ou sélectionné une photo de la librairie

Arg1 : pointeur vers UIImagePickerController créé précédemment

Arg2: instance de UIImage contenant la photo sélectionnée

Arg3 : instance de NSDictionary pour l'édition

le dictionnaire contient l'image originale sous la clé :

UIImagePickerControllerOriginalImage

imagePickerControllerDidCancel:

Est appelée quand Cancel. L'utilisation de Image picker est terminé

Au minimum

```
- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker {  
    [picker dismissModalViewControllerAnimated:YES];  
}
```

Appareil photo et librairie

Implémentation du délégué du contrôleur de Image Picker

Protocole UIImagePickerControllerDelegate

imagePickerController:didFinishPickingImage:editingInfo:

Arg1 : pointeur vers UIImagePickerController créé précédemment

Arg2: instance de UIImage contenant la photo sélectionnée

Arg3 : instance de NSDictionary pour l'édition

le dictionnaire contient l'image originale sous la clé :

UIImagePickerControllerOriginalImage

Recherche de l'image originale

```
- (void)imagePickerController:(UIImagePickerController *)picker  
    didFinishPickingImage:(UIImage *)image  
    editingInfo:(NSDictionary *)editingInfo {  
  
    UIImage *selectedImage = image;  
    UIImage *originalImage = [editingInfo objectForKey:  
    UIImagePickerControllerOriginalImage];  
  
    // do something with selectedImage and originalImage  
  
    [picker dismissModalViewControllerAnimated:YES];  
}
```

Appareil photo et librairie

Implémentation du délégué du contrôleur de Image Picker

Protocole UIImagePickerControllerDelegate

imagePickerController:didFinishPickingImage:editingInfo:

Arg1 : pointeur vers UIImagePickerController créé précédemment

Arg2: instance de UIImage contenant la photo sélectionnée

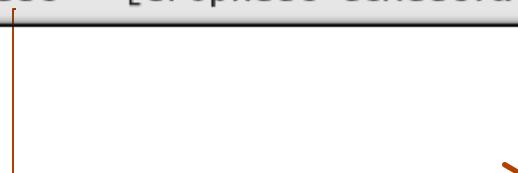
Arg3 : instance de NSDictionary pour l'édition

le dictionnaire contient l'image originale sous la clé :

UIImagePickerControllerOriginalImage

Choix d'une zone d'intérêt et conversion de la chaîne vers un CGRect

```
NSValue *cropRect = [editingInfo  
objectForKey:UIImagePickerControllerCropRect];  
CGRect theRect = [cropRect CGRectValue];
```



Appareil photo et librairie

Exemple de programme : view-based application template

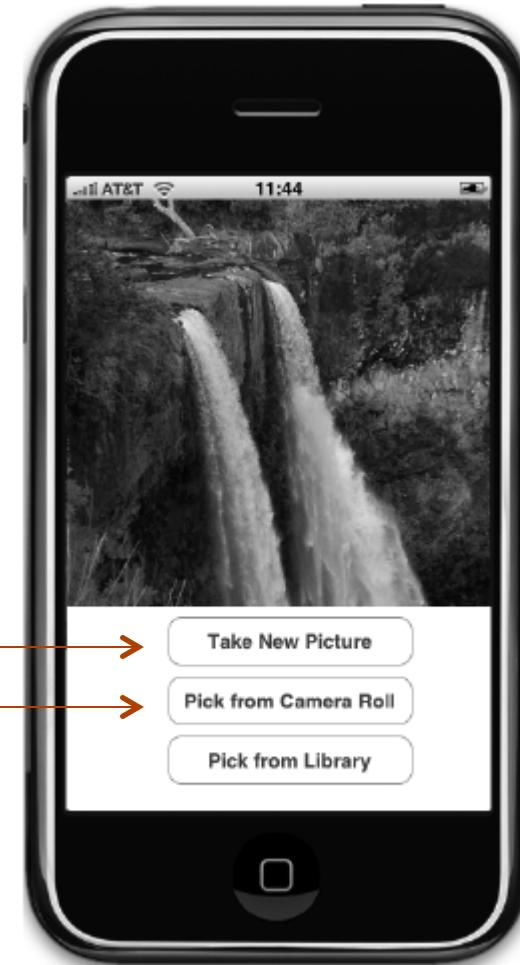
Outlets :

- pointer sur image view pour mettre à jour l'image provenant de image picker.

- pointer vers les boutons *IBOutlet*

Actions :

- pour les boutons TNP et PfCR
- pour PfL



Application caméra

Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.h

```
#import <UIKit/UIKit.h>

@interface CameraViewController : UIViewController
    <UIImagePickerControllerDelegate, UINavigationControllerDelegate> {
    UIImageView *imageView;
    UIButton *takePictureButton;
    UIButton *selectFromCameraRollButton;
}
@property (nonatomic, retain) IBOutlet UIImageView *imageView;
@property (nonatomic, retain) IBOutlet UIButton *takePictureButton;
@property (nonatomic, retain) IBOutlet UIButton
    *selectFromCameraRollButton;
- (IBAction)getCameraPicture:(id)sender;
- (IBAction)selectExistingPicture;
@end
```

2 protocoles :
Sous classe de UINavigationControllerDelegate

Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.xib (Interface Builder)

3 boutons Round Rect Buttons → View

Ecrire les intitulés des boutons

Lien vers l'icône File's Owner → *takePictureButton*

⌘2: Touch Up Inside → File's Owner → getCameraPicture:

→ *selectFromCamera*

RollButton

⌘2 : Touch Up Inside → File's Owner → getCameraPicture:

→ *Pick from Library*

⌘2 : Touch Up Inside → File's Owner → selectExistingPicture

1 Vue image

Lien vers l'icône File's Owner → outlet imageView



Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.m

```
#import "CameraViewController.h"

@implementation CameraViewController
@synthesize imageView;
@synthesize takePictureButton;
@synthesize selectFromCameraRollButton;

- (void)viewDidLoad {
    if (!UIImagePickerController isSourceTypeAvailable:
        UIImagePickerControllerSourceTypeCamera]) {
        takePictureButton.hidden = YES;
        selectFromCameraRollButton.hidden = YES;
    }
}
```

```
...
- (void)viewDidUnload {
    // Release any retained subviews of the main view.
    // e.g. self.myOutlet = nil;
    self.imageView = nil;
    self.takePictureButton = nil;
    self.selectFromCameraRollButton = nil;
    [super viewDidUnload];
}

- (void)dealloc {
    [imageView release];
    [takePictureButton release];
    [selectFromCameraRollButton release];
    [super dealloc];
}
...
```

Vérification si le mobile possède un appareil photo
Et cachons éventuellement 2 des 3 boutons.

Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.m

```
...
#pragma mark -
- (IBAction)getCameraPicture:(id)sender {
    UIImagePickerController *picker =
        [[UIImagePickerController alloc] init];
    picker.delegate = self;
    picker.allowsImageEditing = YES;
    picker.sourceType = (sender == takePictureButton) ?
        UIImagePickerControllerSourceTypeCamera :
        UIImagePickerControllerSourceTypeSavedPhotosAlbum;
    [self presentModalViewController:picker animated:YES];
    [picker release];
}
```

Allocation et initialisation d'une
Instance UIImagePickerController

Délégué ← self
Permission pour l'édition
Choix de l'action

Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.m (suite)

```
- (IBAction)selectExistingPicture {
    if ([UIImagePickerController isSourceTypeAvailable:
        UIImagePickerControllerSourceTypePhotoLibrary]) {
        UIImagePickerController *picker =
            [[UIImagePickerController alloc] init];
        picker.delegate = self;
        picker.sourceType = UIImagePickerControllerSourceTypePhotoLibrary;
        [self presentModalViewController:picker animated:YES];
        [picker release];
    }
    else {
        UIAlertView *alert = [[UIAlertView alloc]
            initWithTitle:@"Error accessing photo library"
            message:@"Device does not support a photo library"
            delegate:nil
            cancelButtonTitle:@"Drat!"
            otherButtonTitles:nil];
        [alert show];
        [alert release];
    }
}
```

Création d'un image picker avec
un sourcetype de *UIImagePickerControllerSourceTypePhotoLibrary*

Alerte si le mobile n'a pas de
Librairie photo

Appareil photo et librairie

Exemple de programme : view-based application template

CameraViewController.m (suite / les méthodes déléguées)

Affiche l'image retournée
(imageView)

```
#pragma mark -
- (void)imagePickerController:(UIImagePickerController *)picker
    didFinishPickingImage:(UIImage *)image
    editingInfo:(NSDictionary *)editingInfo {
    imageView.image = image;
    [picker dismissModalViewControllerAnimated:YES];
}
- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker {
    [picker dismissModalViewControllerAnimated:YES];
} @end
```

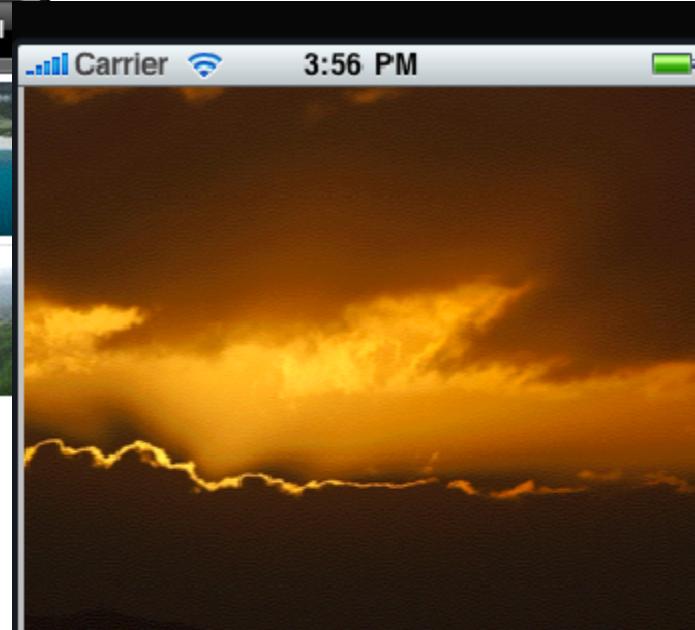
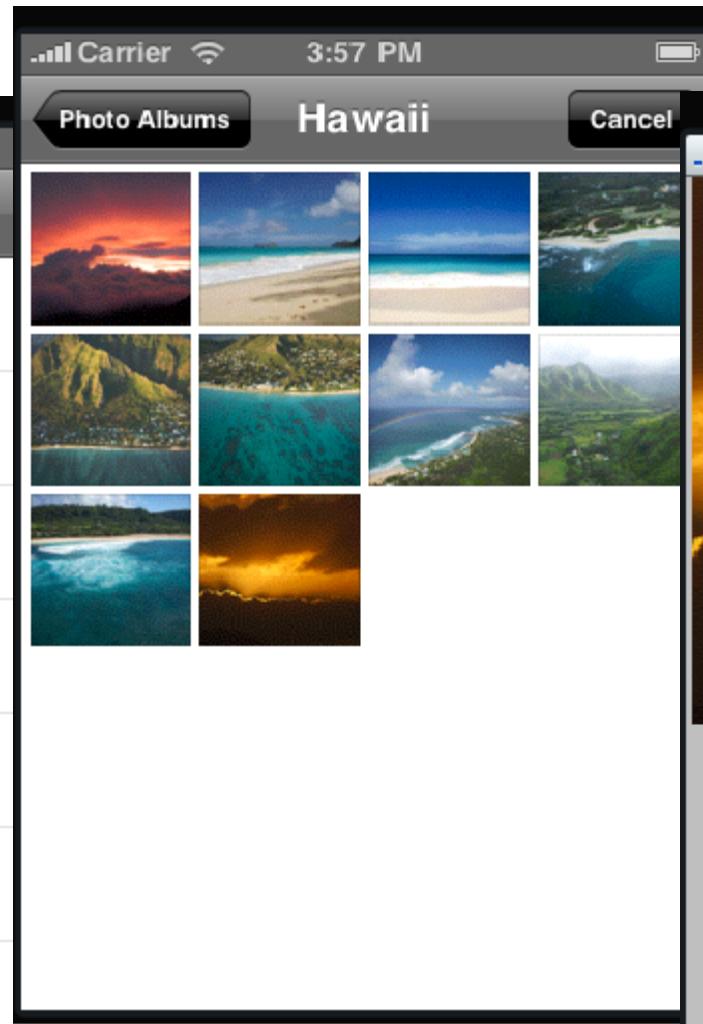
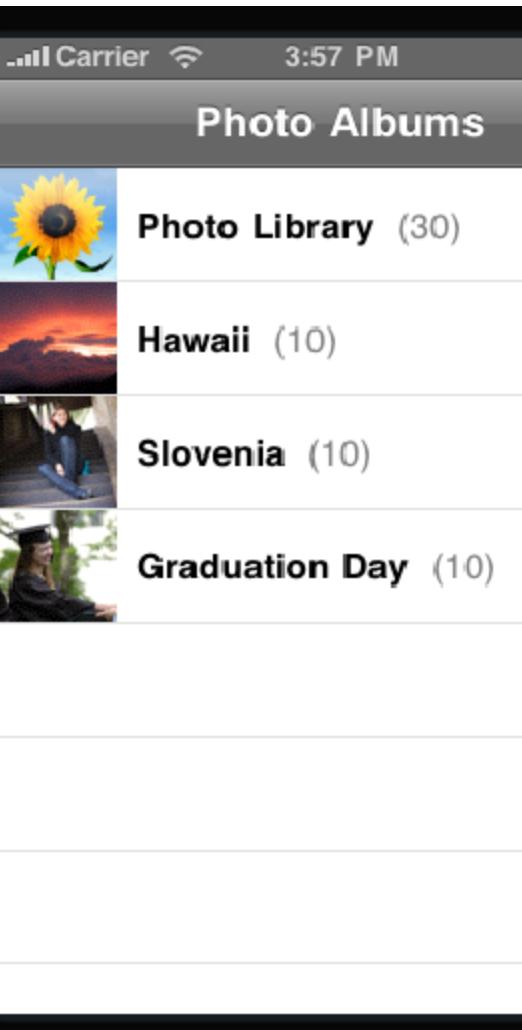


Et retour sur l'application
après déconnexion

Appareil photo et librairie

Exemple de programme : view-based application template

Simulation

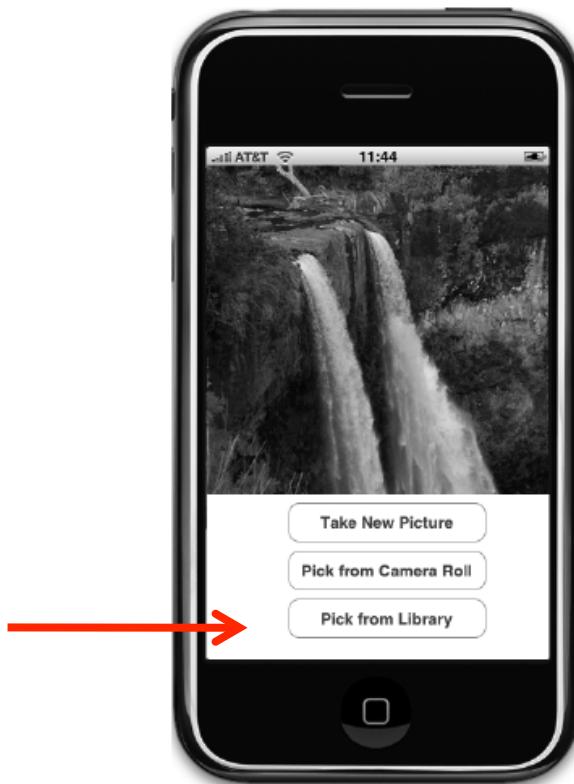


Pick From Library

Appareil photo et librairie

Exemple de programme : view-based application template

Simulation



Exécution sur un Iphone



Possibilité de zoomer et
de sélectionner une zone
d'intérêt