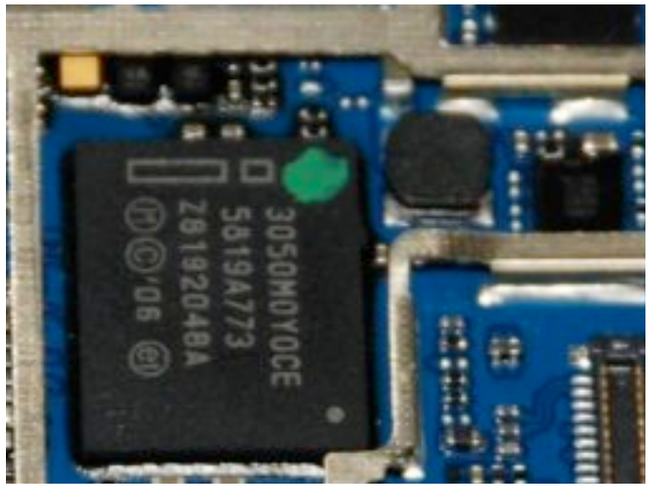
# Advanced Mobile Solution
# AMS

## Cours 3
## Persistence sur iPhone

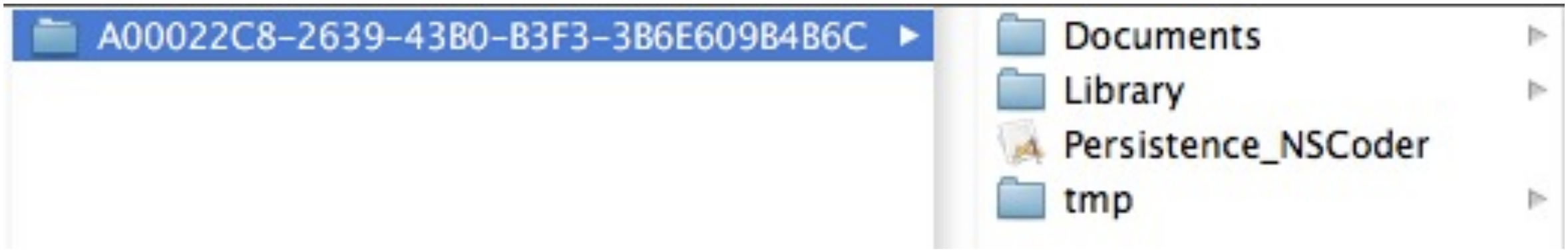


# Sommaire: persitence sur *iPhone*

- **Structure des fichiers**
- **Simple écriture/lecture dans un fichier**
- **Property list (.plist) +** Appli. settings & User Pref.
- **Archivage clé-valeur** & **NSCoding**
- ~~**SQLite via Core Data**~~
- ~~**SQLite via SQL**~~

# Structure des fichiers

**/Users/wagen/Library/Application Support/iPhone Simulator/4.1/Applications/**
**A00022C8-2639-43B0-B3F3-3B6E609B4B6C/Documents/me.txt**




**/var/mobile/Applications/6117C930-F5AE-4C9C-8AD0-74D098C1EB0F/Documents/me.txt**

***Chemin_par_défaut* /me.txt**

```
– (NSString *) documentDirectoryPath
{
    NSArray *paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,
    NSUserDomainMask, YES);

    return [paths objectAtIndex: 0];
}
```

***Chemin par défaut***

# Creating Paths and Locating Directories

# Exemple 1

**/var/mobile/Applications/6117C930-F5AE-4C9C-8AD0-74D098C1EB0F/Documents/MyFile**

**Chemin_par_défaut /MyFile**

```
NSArray *paths = NSSearchPathForDirectoriesInDomains(
                    NSDocumentDirectory,
                    NSUserDomainMask, YES);
```

```
NSString *theCompletePath =
 [[paths objectAtIndex: 0] stringByAppendingPathComponent: @"MyFile"];
```

# Exemple 2

```
NSString *path2 = [NSHomeDirectory()
 stringByAppendingPathComponent:@"Documents/MyFile2.txt"];
```

# writeToFile: / stringWithContentsOfFile:

## écriture (NSString *stringToBeSaved = @"Voilà un texte à sauvegarder !")

```
[stringToBeSaved writeToFile: aFilePath atomically: YES encoding: NSUTF8StringEncoding error: NULL];
```

## lecture

```
NSString *stringToBeRestored = [NSString stringWithContentsOfFile: aFilePath encoding: NSUTF8StringEncoding error: NULL];
```

## Parfait pour stocker des données "simples" (NSString)

```
@interface Person : NSObject
{
  NSString *name;
  NSDate *birthDate;
  float weight;
  BOOL married;
}
```

## ... mais "inventer" votre propre format

```
NSString *concatenatedPerson = [NSString stringWithFormat: @"%@|%@|%f|%d",
name, [[self dateFormatter] stringFromDate: birthDate], weight, married];
```

## voir exercice

# Persistence_File.xcodeproj

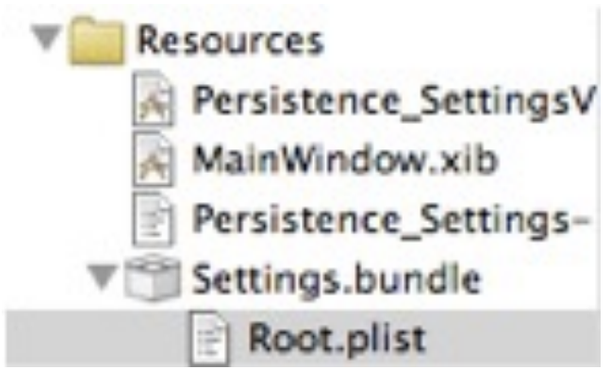
# Exercice:

Persistence_File.xcodeproj

# Property list (.plist)

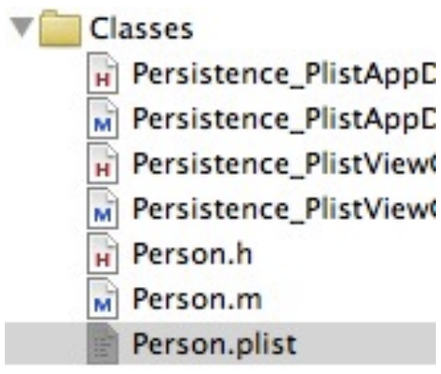
contient des objets correspondants à des clés. Ces fichiers sont au format XML et sont utilisés pour 2 fonctionnalités différentes :

- Gestion des préférences utilisateur (=Application settings, user default, ...)

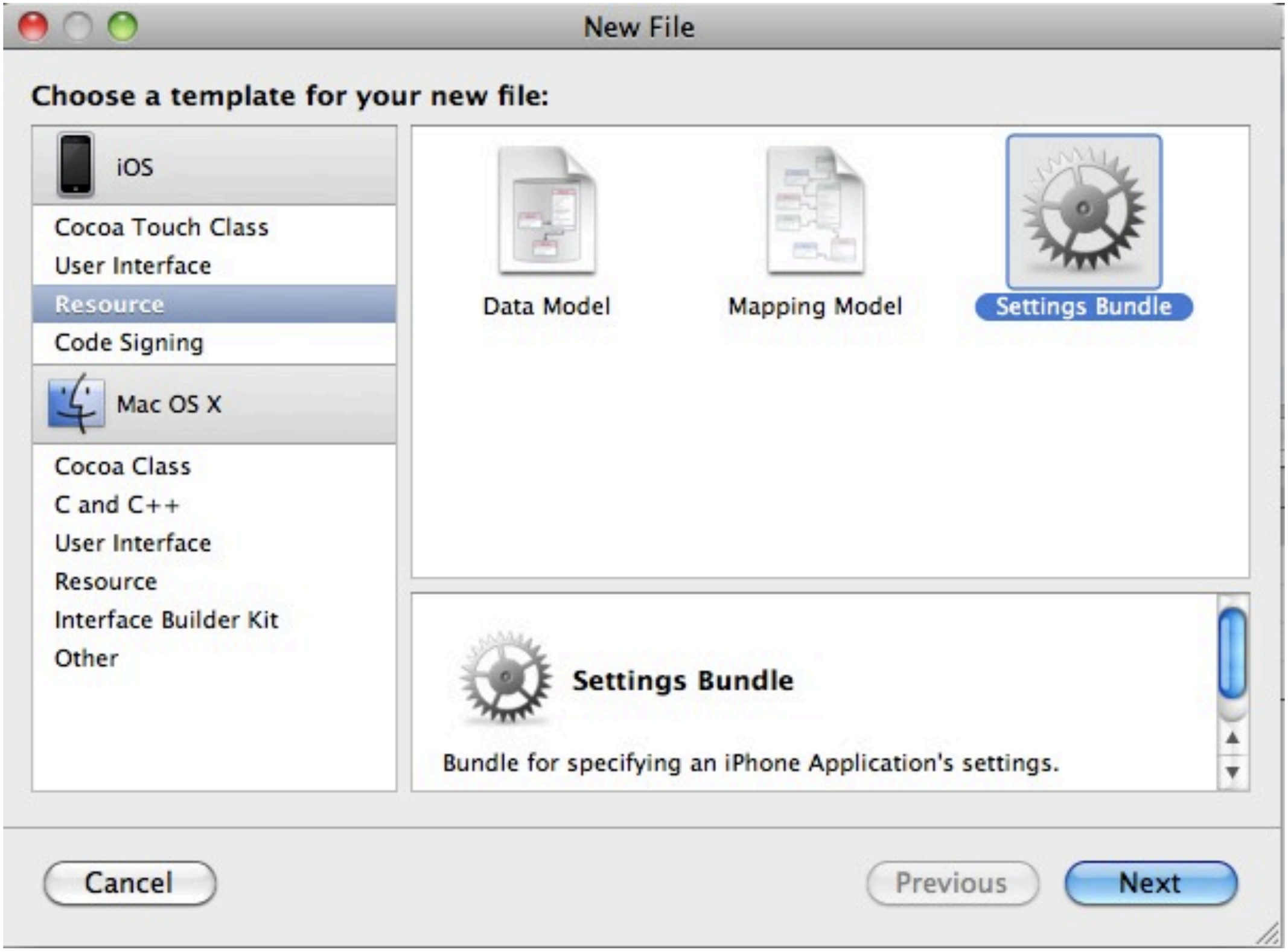




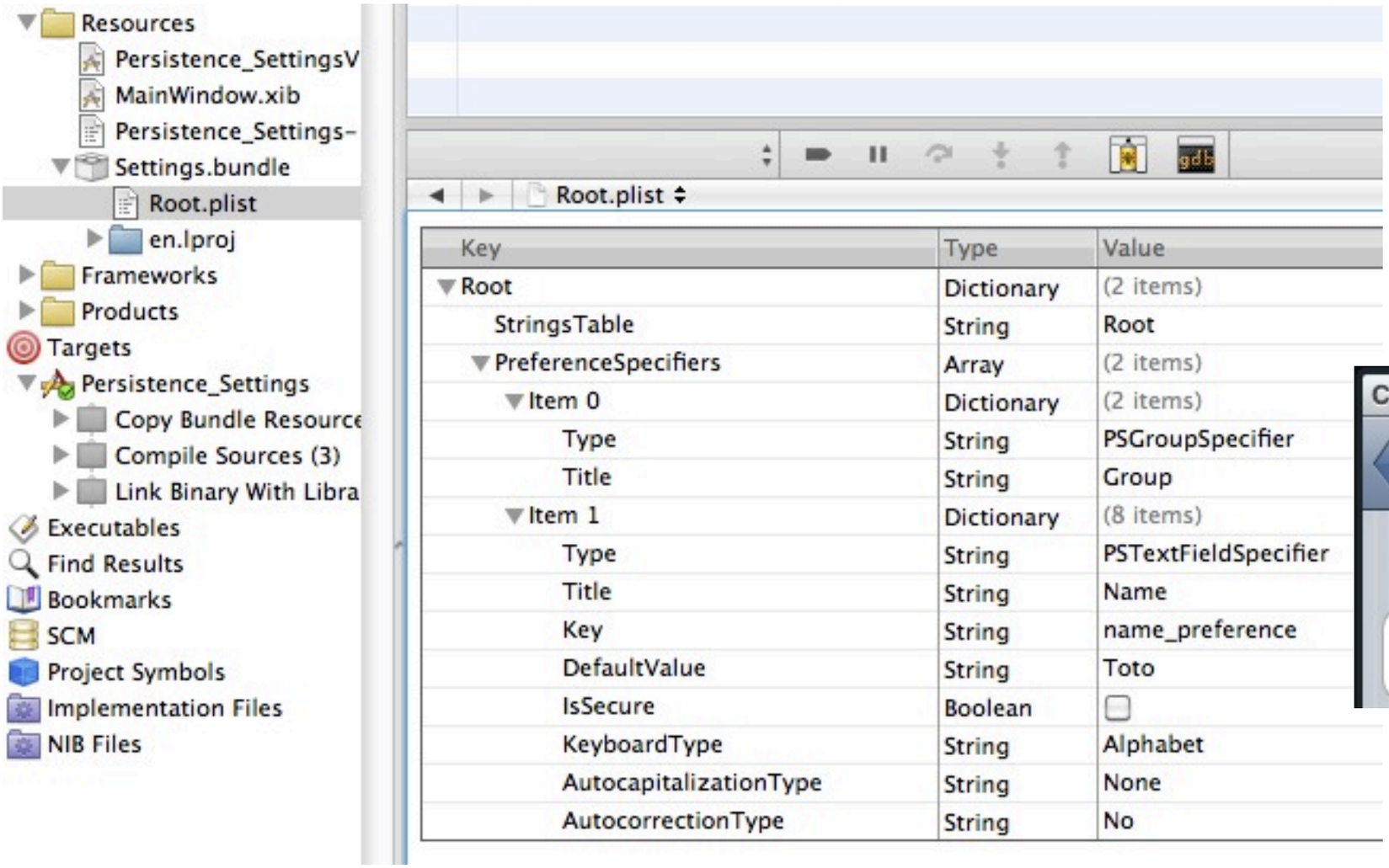
- Stockage d'information dans son propre fichier .plist



# Ajouter un Settings Bundle



# Ajouter un Settings.bundle




# Ecriture/lecture Settings

- **Ref: Property List Programming Guide**

## écriture


A defaults database is created automatically for each user by `standardUserDefaults`.

```
[[NSUserDefaults standardUserDefaults]
                setValue: @"Tototest" forKey: @"Name_preference"];


[[NSUserDefaults standardUserDefaults] synchronize];
```

## lecture

```
[[NSUserDefaults standardUserDefaults] stringForKey: @"Name"]
```

```objc
- (void) manageUserDefaults
{
   NSString *pathStr = [[NSBundle mainBundle] bundlePath];
   NSString *settingsBundlePath = [pathStr stringByAppendingPathComponent:
@"Settings.bundle"];
   NSString *finalPath = [settingsBundlePath stringByAppendingPathComponent:
@"Root.plist"];

   NSDictionary *settingsDict = [NSDictionary dictionaryWithContentsOfFile: finalPath];
   NSArray *prefSpecifierArray = [settingsDict objectForKey: @"PreferenceSpecifiers"];

   NSMutableDictionary *appDefaults = [NSMutableDictionary dictionary];
   NSString *nameDefaultValue;

   NSDictionary *prefItem;

   for (prefItem in prefSpecifierArray)
   {
       NSString *keyValueStr = [prefItem objectForKey: @"Key"];
       id defaultValue = [prefItem objectForKey: @"DefaultValue"];

          // Retrieve the default value for name_preference (only at first launch)
       if ([keyValueStr isEqualToString: @"name_preference"] &&
       ([[NSUserDefaults standardUserDefaults] stringForKey: @"name_preference"] == nil))
       {
          nameDefaultValue = defaultValue;
          [appDefaults setObject: nameDefaultValue forKey: @"name_preference"];
       }
   }

       // Set the settings in the UserDefaults and synchronize
   [[NSUserDefaults standardUserDefaults] registerDefaults: appDefaults];
   [[NSUserDefaults standardUserDefaults] synchronize];
}
```

# Persistence pour des objets

- Archivage “Model Objects”, ou “NSCoding” ou “clé-valeur”

# Archivage clé-valeur & NSCoding

- Pour archiver *NSKeyedArchiver*

```
[NSKeyedArchiver archiveRootObject: objetAArchiver toFile: [self filePath]];
```

- Pour restorer *NSKeyedUnarchiver*

```
objetARestorer = (NSObject *)[NSKeyedUnarchiver unarchiveObjectWithFile: [self filePath]];
```

Seule contrainte :
objetAArchiver et *objetARestorer*
doivent être conforme à NSCoding

```
@interface objetAArchiver : NSObject <NSCoding>
```

# Déclaration de l'objet Person (fichier Person.h)

```
//  Person.h

#import <Foundation/Foundation.h>

@interface Person : NSObject <NSCoding>
{
  NSString *name;
  NSDate *birthDate;
  float weight;
  BOOL married;
}

@property (nonatomic, retain) NSString *name;
@property (nonatomic, retain) NSDate *birthDate;
@property (nonatomic) float weight;
@property (nonatomic) BOOL married;

- (id) initWithName: (NSString *)aName
          birthDate: (NSDate *)aBirthDate
             weight: (float)aWeight
          isMarried: (BOOL)aStatus;

@end
```

# Implémentation de l'objet Person (fichier Person.m)

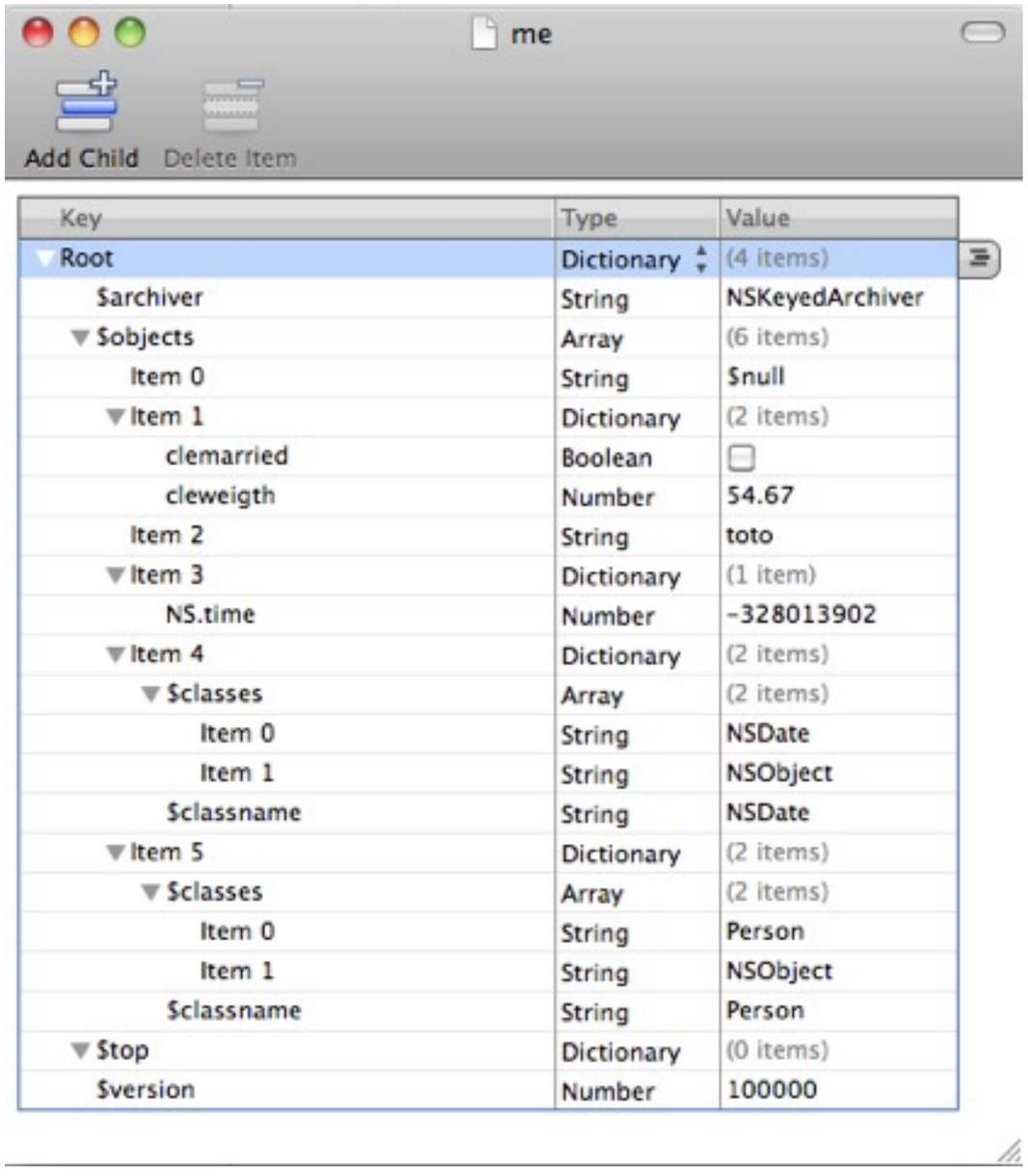
```
//  Person.m

#import "Person.h"

@implementation Person

@synthesize name, birthDate, weight, married;


- (void) encodeWithCoder: (NSCoder *)encoder
{
    [encoder encodeObject: name forKey: @"clename"];
    [encoder encodeObject: birthDate forKey: @"clebirthDate"];
    [encoder encodeFloat: weight forKey: @"cleweigth"];
    [encoder encodeBool: married forKey: @"clemarried"];
}

- (id) initWithCoder: (NSCoder *)decoder
{
    self = [super init];

    [self setName: [decoder decodeObjectForKey: @"clename"]];
    [self setBirthDate: [decoder decodeObjectForKey: @"clebirthDate"]];
    [self setWeight: [decoder decodeFloatForKey: @"cleweigth"]];
    [self setMarried: [decoder decodeBoolForKey: @"clemarried"]];

    return self;
}

// Constructeur
- (id) initWithName: (NSString *)aName birthDate: (NSDate *)aBirthDate weight: (float)aWeight isMarried: (BOOL)
aStatus
{
    self = [super init];

    [self setName: aName];
    [self setBirthDate: aBirthDate];
    [self setWeight: aWeight];
    [self setMarried: aStatus];

    return self;
}

@end
```

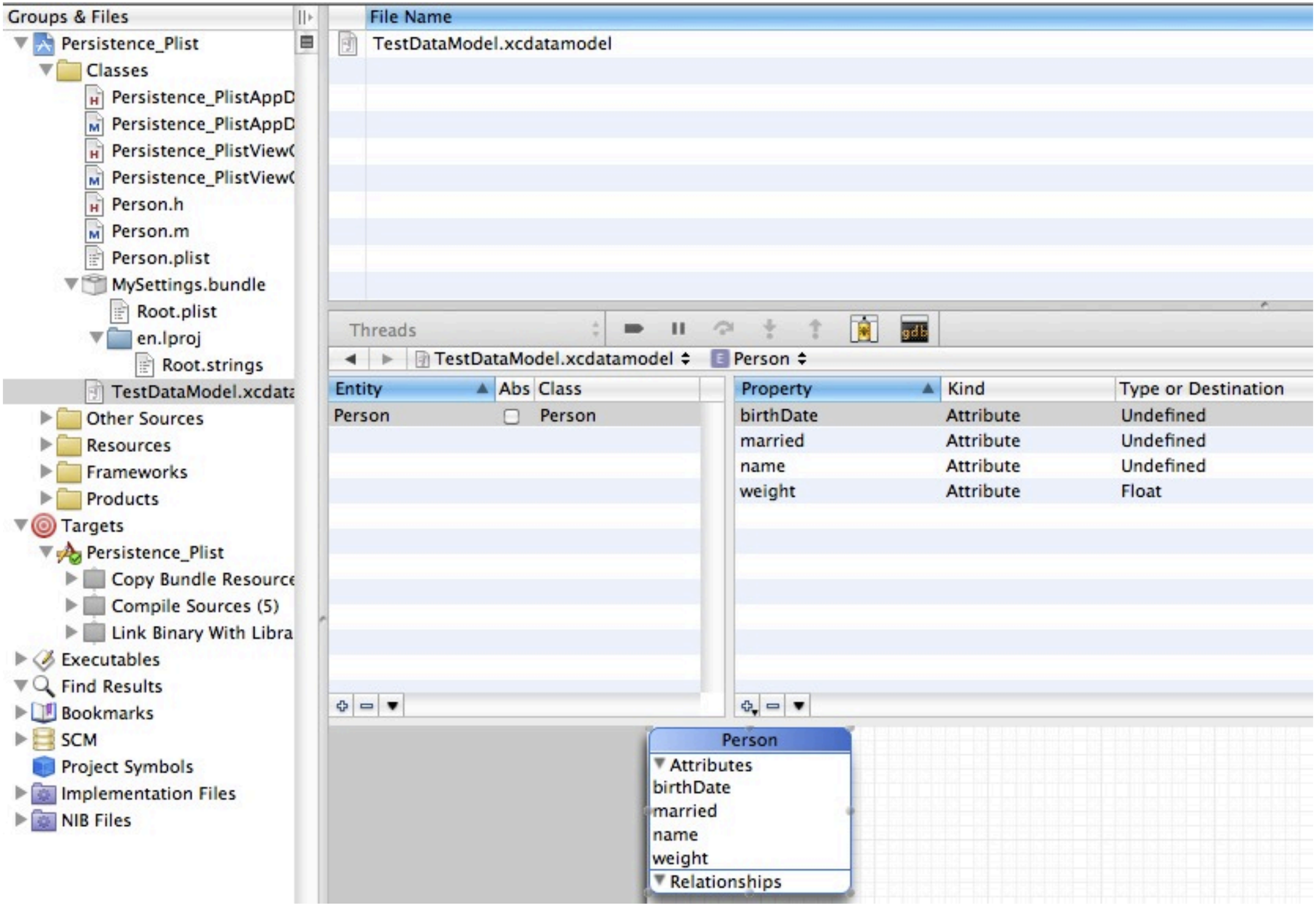
# Exercices

# fichier *output.plist*

## écriture

```
[array writeToFile:@"a_valid_path/output.plist" atomically:YES];
```




Groups & Files
Persistence_Plist
Classes
Persistence_PlistAppD
Persistence_PlistAppD
Persistence_PlistView(
Persistence_PlistView(
Person.h
Person.m
Person.plist
MySettings.bundle
Root.plist
en.lproj
Root.strings
TestDataModel.xcdata
Other Sources
Resources
Frameworks
Products
Targets
Persistence_Plist
Copy Bundle Resource
Compile Sources (5)
Link Binary With Libra
Executables
Find Results
Bookmarks
SCM
Project Symbols
Implementation Files
NIB Files
File Name
TestDataModel.xcdatamodel
Threads
TestDataModel.xcdatamodel
Person
Entity
Abs
Class
Person
Person
Property
Kind
Type or Destination
birthDate
Attribute
Undefined
married
Attribute
Undefined
name
Attribute
Undefined
weight
Attribute
Float
Person
Attributes
birthDate
married
name
weight
Relationships