

BASE DE DONNÉES MOBILES

CORE DATA
COURS IUT LYON 1

PLAN DE COURS

- Introduction générale à la persistance de données
- Persistance avec des fichiers
- Premiers pas avec Core Data
- Core Data et MVC
- Debug avec Core Data
- Programmation avancée avec Core Data

NOTATION

- Orale
 - Soutenance : **10/20**
 - Participation : **10/20**
- Projet
 - Partie 1 : **5/20**
 - Partie 2 : **5/20**
 - Qualité du code : **5/20**
 - UI & UX : **5/20**

NOTATION - CRITÈRE

- Mise en place des éléments vus en cours
- Structure du projet
- Commentaires, syntaxe, conventions, clarté
- Rapport de projet
- Date de livraison et soutenance 6 Mai

PERSISTANCE DE DONNÉES

- Pourquoi est-ce important ?
- Cas d'utilisation
- Types de persistance
 - Fichiers (Binaires, XML, JSON, Sérialisation, Images, Sons...)
 - User Defaults (Fichiers de propriétés)
 - SQLite
 - iCloud
 - Keychain (Trousseau de clé)
 - CoreData

PERSISTANCE AVEC DES FICHIERS

- À quelle moment il faut utiliser ce type de persistance ?
- Avantages
 - Simple à gérer
 - Possibilité de manipulation des fichiers de taille importante
- Inconvénients
 - Accès lent
 - Pas de relations entre les données
 - Pas de recherche
- Démonstration

KEYCHAIN

- Cas d'utilisation
- Avantages / Inconvénients
- Conseils
- Sauvegarde de données sensibles

CORE DATA

- Base de données
 - Stockage de données en masse
 - Orienté objet
- Core data
 - Base de données orientée objet
 - Framework très performant
 - Surcouche SQLite / XML
- Comment ça marche ?
 - Créer un modèle de données
 - Créer des données et requêtes de base de données avec l'API
 - Accéder aux colonnes de la base via les propriétés

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

Run Stop Master

Enterprise 2 targets, iOS: Enterprise AppDelete AppDelegate Main.storyboard MasterViewDetailViewController.h MasterViewDetailViewController.m DetailViewDetailViewController.h DetailViewDetailViewController.m Images Support Enterprise Framework Products

New Add Files to “Enterprise”... Open... Open Recent Open Quickly... Close Window Close Tab Close “MasterViewController.m” Close Project Save Duplicate... Revert to Saved... Unlock... Export... Show in Finder Open with External Editor Save As Workspace... Project Settings... Create Snapshot... Restore Snapshot... Page Setup... Print...

Tab Window File... Target... Project... Workspace... Group Group from Selection

Today at 19:45 No Issues

Editor View

Créer le modèle de données

No Selection

```
MasterViewController.h
MasterViewController () {
    NSArray *_objects;
}

MasterViewController
FromNib;
fromNib];

load
.dLoad];
additional setup after loading the view, typically from a nib.
.onItem.leftBarButtonItem = self.editButtonItem;

item * addButton = [[UIBarButtonItem alloc] initWithBarButtonSystemItem:UIBarButtonSystemItemAdd target:self action:@selector(insertNewObject:)];
.onItem.rightBarButtonItem = addButton;
```

```
35 - (void) didReceiveMemoryWarning
36 {
37     [super didReceiveMemoryWarning];
38     // Dispose of any resources that can be recreated.
39 }
40
41 - (void)insertNewObject:(id)sender
42 {
43     if (!_objects) {
44         _objects = [[NSMutableArray alloc] init];
45     }
46     [_objects insertObject:[NSDate date] atIndex:0];
47     NSIndexPath *indexPath = [NSIndexPath indexPathForRow:0 inSection:0];
48     [self.tableView insertRowsAtIndexPaths:@[indexPath] withRowAnimation:UITableViewRowAnimationAutomatic];
49 }

#pragma mark - Table View
50
51 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
52 {
53     return 1;
54 }
55
56 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
57 {
58     return _objects.count;
59 }
60
61 }
```

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

MasterViewController.m

Choose a template for your new file:

iOS

- Cocoa Touch
- C and C++
- User Interface
- Core Data
- Resource
- Other

OS X

- Cocoa
- C and C++
- User Interface
- Core Data
- Resource
- Other

Data Model

Mapping Model

NSManagedObject subclass

A Core Data model file that allows you to use the design component of Xcode.

Cancel Previous Next

self action:

Ce template

No Selection

```
// MasterViewController.h
// Entrepise
// Created by ... on ...
// Copyright (c) 2013 ...
// All rights reserved.

#import "MasterViewController.h"
#import "DetailViewController.h"

@interface MasterViewController : UITableViewController<UISearchBarDelegate>
@property (strong, nonatomic) NSMutableArray *_objects;
@end

@implementation MasterViewController
- (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.
}

- (void)insertNewObject:(id)sender
{
    if (!_objects) {
        _objects = [[NSMutableArray alloc] init];
    }
    [_objects insertObject:[NSDate date] atIndex:0];
    NSIndexPath *indexPath = [NSIndexPath indexPathForRow:0 inSection:0];
    [self.tableView insertRowsAtIndexPaths:@[indexPath] withRowAnimation:UITableViewRowAnimationAutomatic];
}

#pragma mark - Table View

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
    return _objects.count;
}
```

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Entrepise > iPhone Retina (3.5-inch)

Entrepise: Ready | Today at 19:45

No Issues

Run Stop Scheme

MasterViewController.m

Entrepise

Enterprise

- AppDelegate.h
- AppDelegate.m
- Main.storyboard
- MasterViewController.h
- MasterViewController.m**
- DetailViewController.h
- DetailViewController.m
- Images.xcassets
- Supporting Files
- EnterpriseTests
- Frameworks
- Products

Choose a template for Save As: Model

Tags:

Favorites

- William
- Applications
- Desktop
- Documents
- Downloads
- Dropbox
- Dropbox

Enterprise

- AppDelegate.h
- Enterprise.xcodeproj
- EnterpriseTests
- Base.iproj
- DetailViewController.h
- DetailViewController.m
- en.iproj
- Entrepise-Info.plist
- Entrepise-Prefix.pch
- Images.xcassets
- main.m

No Selection

Group: Enterprise

Targets: Entrepise EnterpriseTests

New Folder

Create

Cancel

self action:

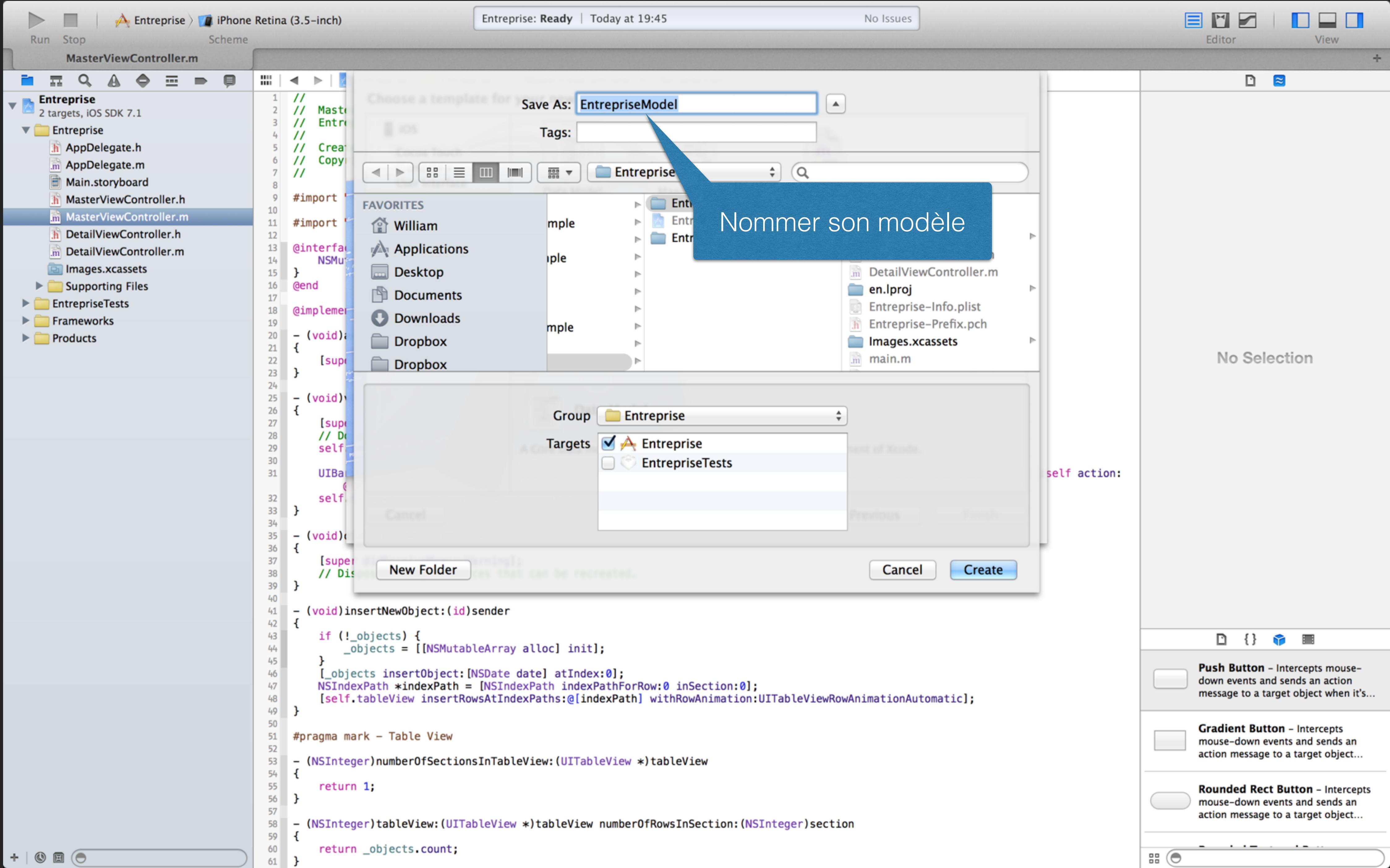
Push Button - Intercepts mouse-down events and sends an action message to a target object when it's...

Gradient Button - Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button - Intercepts mouse-down events and sends an action message to a target object...

```

1 // 
2 // MasterViewController.h
3 // Enterprise
4 // 
5 // Create
6 // Copy
7 // 
8 #import "MasterViewController.h"
9 #import "DetailViewController.h"
10 @interface MasterViewController : UITableViewController<DetailViewControllerDelegate>
11 @end
12 @implementation MasterViewController
13 - (void)viewDidLoad {
14     [super viewDidLoad];
15 }
16 - (void)didReceiveMemoryWarning {
17     [super didReceiveMemoryWarning];
18     // Dispose of any resources that can be recreated.
19 }
20 - (void)insertNewObject:(id)sender {
21     if (!_objects) {
22         _objects = [[NSMutableArray alloc] init];
23     }
24     [_objects insertObject:[NSDate date] atIndex:0];
25     NSIndexPath *indexPath = [NSIndexPath indexPathForRow:0 inSection:0];
26     [self.tableView insertRowsAtIndexPaths:@[indexPath] withRowAnimation:UITableViewRowAnimationAutomatic];
27 }
28 #pragma mark - Table View
29 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
30 {
31     return 1;
32 }
33 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
34 {
35     return _objects.count;
36 }
37 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
38 {
39     static NSString *CellIdentifier = @"Cell";
40     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
41     if (cell == nil) {
42         cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:CellIdentifier];
43     }
44     NSDate *date = [_objects objectAtIndex:indexPath.row];
45     cell.textLabel.text = [date description];
46     return cell;
47 }
48 - (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPath
49 {
50     DetailViewController *detailViewController = [[DetailViewController alloc] initWithNibName:@"DetailViewController" bundle:nil];
51     detailViewController.date = [_objects objectAtIndex:indexPath.row];
52     [self.navigationController pushViewController:detailViewController animated:YES];
53 }
54 
```



EntrepiseModel.xcdatamodel

Enterprise 2 targets, iOS SDK 7.1

Enterprise

- AppDelegate.h
- AppDelegate.m
- Main.storyboard
- MasterViewController.h
- MasterViewController.m
- EntrepiseModel.xcdatamodeld A
- DetailViewController.h
- DetailViewController.m
- Images.xcassets
- Supporting Files
- EnterpriseTests
- Frameworks
- Products

ENTITIES

FETCH REQUESTS

CONFIGURATIONS

Default

Entities

Entity Abstract Class

Quick Help

No Quick Help

Notre nouveau modèle

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

Gradient Button - Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button - Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity Add Attribute Editor Style

Entrepise iPhone Retina (3.5-inch)

Entreprise: Ready | Today at 19:45

No Issues

Run Stop Scheme

EntrepriseModel.xcdatamodel

ENTITIES Entity

ATTRIBUTES

Relationships

Fetched Properties

Attributs

Ajouter une entité

Editor View

Quick Help No Quick Help

Add Entity

Add Attribute

Editor Style

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

Gradient Button - Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button - Intercepts mouse-down events and sends an action message to a target object...

Nommer l'entité
Ex : Employee

Ajouter un attribut

Ajouter un attribut

EnterpriseModel.xcdatamodel

ENTITIES

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Attribute ▲ Type

+ -

Relationships ▲ Destination Inverse

+ -

Fetched Properties ▲ Predicate

+ -

Add Entity Add Attribute Editor Style

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

EntrepriseModel.xcdatamodel

ENTITIES Employee

ATTRIBUTES attribute Type Undefined

Relationships

Fetched Properties

Quick Help No Quick Help

Nom de l'attribut

Type de l'attribut

Erreur : Un attribut sans type

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

Gradient Button - Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button - Intercepts mouse-down events and sends an action message to a target object...

EntrepriseModel.xcdatamodel

ENTITIES Employee

FETCH REQUESTS

CONFIGURATIONS Default

Attributes

Attribute ▲	Type
N accessCode	Integer 16
S photo	String
D birthdate	Date
D dateOfEntry	Date
S firstname	String
S lastname	String
N salary	Double

+ -

Relationships

Relationship ▲	Destination	Inverse

+ -

Fetched Properties

Fetched Property ▲

+ -

Quick Help
No Quick Help

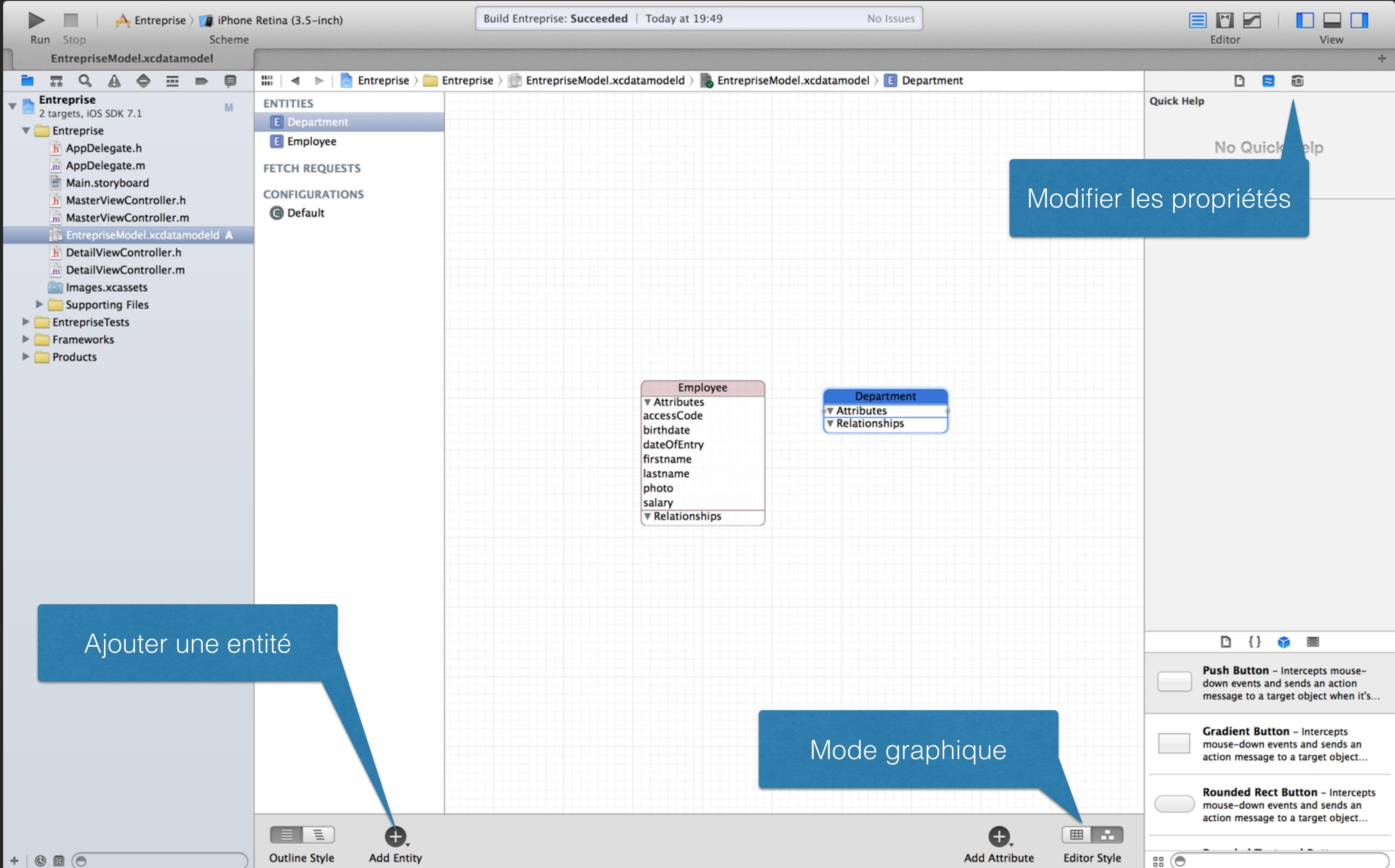
Push Button – Intercepts mouse-down events and sends an action message to a target object when it's...

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity Add Attribute Editor Style

Il est possible de définir toute sorte de types d'attributs
Tous les attributs sont des objets Objective-C : NSString, NSData, NSSet, NSDate, NSNumber



Run Stop

Scheme

Editor View

EntrepriseModel.xcdatamodel

ENTITIES

E Department

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Employee

▼ Attributes

accessCode
birthdate
dateOfEntry
firstname
lastname
photo
salary

▼ Relationships

Department

▼ Attributes

attribute

▼ Relationships

Définir les propriétés d'un attribut

Attribute

Name

Properties Transient Optional
 Indexed

Attribute Type

Advanced Index in Spotlight
 Store in External Record File

User Info

Key
Value

+ -

Versioning

Hash Modifier

Renaming ID

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity

Add Attribute Editor Style

Valeur par défaut

The screenshot shows the Xcode Data Model Editor interface. On the left, the Project Navigator displays the project structure for 'Entreprise' with files like AppDelegate.h, Main.storyboard, and DetailViewController.h. The Data Model Navigator shows the 'EntrepiseModel.xcdatamodeld' file with entities 'Department' and 'Employee'. The 'Department' entity is currently selected. In the center, the Entity Inspector shows the 'name' attribute for 'Department' with a placeholder 'Default Value'. A callout bubble points to this field. Below the entities, the Attribute inspector provides detailed settings for the 'name' attribute, including validation rules (No Value, Min Length, Max Length) and optional properties (Optional checked). The Employee entity is also visible with its attributes: accessCode, birthdate, dateOfEntry, firstname, lastname, photo, salary, and Relationships.

ENTITIES

E Department

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Employee

▼ Attributes

accessCode
birthdate
dateOfEntry
firstname
lastname
photo
salary

▼ Relationships

Department

▼ Attributes

name

▼ Relationships

Attribute

Name **name**

Properties Transient Optional
 Indexed

Attribute Type **String**

Validation No Value Min Length
 No Value Max Length

Default Value Default Value

Reg. Ex. Regular Expression

Advanced Index in Spotlight
 Store in External Record File

User Info

Key ▲ Value

+ -

Versioning

Hash Modifier Version Hash Modifier

Renaming ID Renaming Identifier

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity Add Attribute Editor Style

EntrepriseModel.xcdatamodel

ENTITIES

E Department
E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Entity

Name

Class

Abstract Entity

Parent Entity

Indexes

+ -

User Info

Key

+ -

Versioning

Hash Modifier

Renaming ID

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

CMD + Glisser pour définir une relation

Outline Style Add Entity Add Attribute Editor Style

```
graph LR; Employee[Employee] <-->|Relationship| Department[Department]
```

Entreprise > iPhone Retina (3.5-inch)

Build Entreprise: Succeeded | Today at 19:49

No Issues

Run Stop Scheme

Editor View +

EntrepiseModel.xcdatamodel

Entrepise

2 targets, iOS SDK 7.1

Enterprise

AppDelegate.h
AppDelegate.m
Main.storyboard
MasterViewController.h
MasterViewController.m
EntrepiseModel.xcdatamodeld
DetailViewController.h
DetailViewController.m
Images.xcassets

Supporting Files

EntrepiseTests

Frameworks

Products

ENTITIES

E Department
E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Relationship

Name department
Properties Transient Optional
Destination Department
Inverse newRelationship
Delete Rule Nullify
Type To One
 To Many
Advanced Index in Spotlight
 Store in External Record File

User Info

Key Value

+ -

Versioning

Hash Modifier Version Hash Modifier
Renaming ID Renaming Identifier

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity Add Attribute Editor Style

Nommer la relation

Choisir le type de relation

The screenshot shows the Xcode Data Model Editor for an iOS project named 'Entrepise'. The left sidebar lists files like AppDelegate.h, Main.storyboard, and EntrepiseModel.xcdatamodeld. The main canvas displays two entities: 'Employee' and 'Department'. The 'Employee' entity has attributes: accessCode, birthdate, dateOfEntry, firstname, lastname, photo, and salary. It also has a relationship named 'department' pointing to the 'Department' entity. The 'Department' entity has an attribute 'name' and a relationship named 'newRelationship'. A callout bubble labeled 'Nommer la relation' points to the 'Name' field in the 'Relationship' inspector, which is currently set to 'department'. Another callout bubble labeled 'Choisir le type de relation' points to the 'Type' dropdown menu in the same inspector, which is set to 'To One'. The 'Relationship' inspector also includes fields for 'Properties' (with 'Optional' checked), 'Destination' (set to 'Department'), 'Inverse' (set to 'newRelationship'), 'Delete Rule' (nullify), and 'Advanced' options for 'Index in Spotlight' and 'Store in External Record File'.

EntrepriseModel.xcdatamodel

ENTITIES

E Department

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Relationship

Name employees

Properties Transient Optional

Destination Employee

Invert: No Action

Delete Rule: Nullify

Type: Cascade

Deny

Advanced Index in Spotlight Store in External Record File

User Info

Key ▲ Value

+ -

Versioning

Hash Modifier Version Hash Modifier

Renaming ID Renaming Identifier

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Règle de suppression

Attention au cascade !

The screenshot shows the Xcode Data Model Editor. On the left, the project structure for 'Entreprise' is visible, including files like AppDelegate.h, Main.storyboard, and various controller classes. The main workspace displays two entities: 'Employee' and 'Department'. The 'Employee' entity has attributes like accessCode, birthdate, dateOfEntry, firstname, lastname, photo, and salary. It also has a relationship named 'department' with a multiplicity of 1. The 'Department' entity has an attribute 'name' and a relationship named 'newRelationship' with a multiplicity of 1. A blue callout bubble with white text 'Règle de suppression' is positioned above the 'newRelationship' field in the 'Department' entity editor. Another blue callout bubble with yellow text 'Attention au cascade !' is positioned below it. A blue arrow points from the 'Employee' entity towards the 'newRelationship' field. The right side of the screen shows the detailed configuration for the 'newRelationship' field, including options for 'Name' (employees), 'Properties' (with 'Optional' checked), 'Destination' (Employee), and 'Delete Rule' (set to 'Nullify'). Other tabs like 'Relationships', 'Attributes', and 'Configurations' are also visible in the editor.

Entrepise iPhone Retina (3.5-inch)

EntrepriseModel.xcdatamodel

ENTITIES

E Department

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Relationship

Name employees

Properties Transient Optional

Destination Employee

Inverse department

Delete Rule Nullify

Type To Many

Arrangement Ordered

Count Unbounded Minimum
Unbounded Maximum

Advanced Index in Spotlight
 Store in External Record File

User Info

Key Value

+ -

Versioning

Hash Modifier Version Hash Modifier

Renaming ID Renaming Identifier

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

Gradient Button – Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button – Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity Add Attribute Editor Style

Diagram:

```
classDiagram Employee "Employee" <|-- "Attributes": accessCode, birthdate, dateOfEntry, firstname, lastname, photo, salary  
classDiagram Employee "Employee" --> "Relationships": department  
classDiagram Department "Department" <|-- "Attributes": name  
classDiagram Department "Department" --> "Relationships": employees
```

EntrepiseModel.xcdatamodel

ENTITIES

E Department

E Employee

FETCH REQUESTS

CONFIGURATIONS

C Default

Attributes

Attribute ▲	Type
S name	String

+ -

Relationships

Relationship ▲	Destination	Inverse
M employees	Employee	department

+ -

Fetched Properties

Fetched Property ▲	Predicate

+ -

Entity

Name Department

Class NSManagedObject

Abstract Entity

Parent Entity No Parent Entity

Indexes

User Info

Key Value

+ -

Versioning

Hash Modifier Version Hash Modifier

Renaming ID Renaming Identifier

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

Gradient Button - Intercepts mouse-down events and sends an action message to a target object...

Rounded Rect Button - Intercepts mouse-down events and sends an action message to a target object...

Outline Style Add Entity

Add Attribute Editor Style

INTERAGIR AVEC LE MODÈLE

- Le modèle de données est conçu !
- Que faire ?
- Swift ?
- Oui ! Mais où est le code ?

INTERAGIR AVEC LE MODÈLE

- Vous devez disposer d'un **NSManagedObjectContext**
- C'est l'objet central pour communiquer avec la base de données
- Comment l'obtenir ?
 - Créer un **NSManagedObjectContext**
 - Il est également possible d'utiliser un **UIManagedDocument**
 - Le **NSManagedObjectContext** est automatiquement créé pour vous lorsqu'au démarrage du projet, vous cochez "Use Core Data"
 - Démonstration

- Insérer des objets en base

```
let entity = NSEntityDescription.entityForName("Employee", inManagedObjectContext:  
managedContext)
```

```
let employee = NSManagedObject(entity: entity!, insertIntoManagedObjectContext:  
managedContext)
```

- Un NSManagedObject est la représentation objet d'une entrée de la base de données
- Par défaut, tous les attributs de l'objet seront à nil (à moins d'avoir spécifier une valeur par défaut)
- Modifier les attributs de l'objet

```
employee.setValue("William", forKey: "firstname")
```

```
employee.setValue("Antwi", forKey: "lastname")
```

REQUÊTES

- Comment récupérer des objets en base ?
 - Récupérer le **NSManagedObjectContext**
 - Créer un **NSFetchRequest**
 - Exemple

```
let fetchRequest = NSFetchedResultsController(entityName: "Employee")  
var error: NSError?  
  
let fetchedResults = managedObjectContext.executeFetchRequest(fetchRequest, error: &error)  
as [NSManagedObject]?
```

- Démonstration

- Suppression
 - La suppression est très simple
`managedContext.deleteObject(task)`
 - Attention aux règles de suppression
 - Les relations seront automatiquement mises à jour
 - Ne pas garder de référence vers cet objet
- Préparer la suppression
 - Il est possible de préparer un objet à la suppression
 - Pourquoi ?

CORE DATA

- Les classes de Core Data
 - `NSManagedObjectContext`
 - `NSManagedObjectModel`
 - `NSPersistentStoreCoordinator`
 - `NSPersistentStore`
- Data Manager
 - Singleton
 - Gestion centralisée des données

REQUÊTES

- Comment récupérer des objets en base ?
 - Crée un **NSFetchRequest**
 - Définir le nombre d'objets à récupérer
 - Trier ces objets à l'aide **NSSortDescriptors**
 - Filtrer les objets avec un ou plusieurs **NSPredicate**
 - Exemple

```
let fetchedRequest = NSFetchedRequest(entityName: "Task")
fetchedRequest.fetchBatchSize = 20
fetchedRequest.fetchLimit = 100
let sortDescriptor = NSSortDescriptor(key: "name", ascending: true)
fetchedRequest.sortDescriptors = [sortDescriptor]
fetchedRequest.predicate = NSPredicate(format: "name = %@", "Task 1")
```

- **NSSortDescriptor**
 - Le résultat d'une requête est un NSArray the NSManagedObjects
 - Les NSArray sont des tableaux “ordonnés”, il est donc intéressant de trier les données lors des requêtes
 - Il faut donner une list de “sort descriptors” pour décrire le type de tri voulu

```
let sortDescriptor = NSSortDescriptor(key: "name", ascending: true)
fetchedRequest.sortDescriptors = [sortDescriptor]
```
 - Ici tous les object seront triés par l'attribut nom
 - Il est possible de trier par plusieurs attributs
 - Attention à l'ordre de tri

REQUÊTES

- **NSPredicate**
 - Permet de filtrer les objets
 - Langage très puissant qui permet de faire des requêtes complexes
 - Création similaire au **NSString**

```
let taskName = "Task 1"  
let predicate = NSPredicate(format: "name = %@", taskName)
```

- **Exemples**
 - “uniqueId = %@", “1233324”
 - “firstname contains[c] %@", (String)
 - “department.name = %@", (String)

Voir la documentation Apple pour plus d'information

REQUÊTES

- Exécuter la requête
 - Retourne nil en cas d'erreur
 - Retourne un array vide s'il n'y a pas d'objets qui correspondent à la requête
 - Retourne un array de NSManagedObject s'il y a au moins un objet qui correspond
 - Penser à toujours gérer les erreurs

```
var error: NSError?  
  
let fetchedObjects = managedObjectContext.executeFetchRequest(fetchedRequest, error: &error)  
as [Task]?  
  
if let results = fetchedObjects {  
    return results  
} else {  
    println("Could not fetch \(error), \(error?.userInfo)")  
    return nil  
}
```

REQUÊTES

- **Faulting**

- Les objets retournés lors d'une requête ne sont pas forcément en mémoire
- Core Data les charge en mémoire au moment où le programme essaie d'y accéder
- Ceci pour des raisons d'optimisation
- Attention à ceci

```
for task: Task in results {  
    println("Task \$(task)")  
}
```

- Il se pourrait que les attributs apparaissent comme "unfaulted object"
- Pour voir le nom par exemple, il faudra faire

```
for task: Task in results {  
    println("Task \$(task.name)")  
}
```

- Ceci chargera toutes les données

PROJET

- Crée application de gestion de listes de tâches
- Catégorie : Nom, Picto (Prédefinis)
- Tâche : Titre, Description, Photo, Catégorie, Date
- L'application devra au moins permettre de gérer une liste de tâches et de catégorie :
 - Les cellules devra afficher les informations relatives aux tâches (Titre, Description, Photo, Catégorie, Date)
 - Tris : Date, Nom, Catégorie
 - Groupement par catégorie, date ou lettres alphabétiques
 - Il sera possible à l'aide d'un bouton d'accéder à la une vue avec les informations de l'auteur
- Il faudra pouvoir ajouter / supprimer / mettre à jour une catégorie / tâche

PROJET

- Libre à vous d'avoir l'interface que vous voulez
- Attention aux crashes
- Une application fluide est un vrai plus
- Codez proprement
- Laissez place à votre imagination
- William Antwi - wiliam@appslute.fr