Logiciels embarqués ambiants/iOS Chapitre 1 : Prérequis

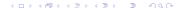
Dr. Abdelkader Gouaïch¹

¹Department of Computer Science Université de Montpellier

2012



- Introduction
 - Objectifs du chaptire
- Prérequis
 - Moteur de jeu
 - Xcode
 - Premier projet sous Xcode
 - Délégation et Protocol
 - Organisation du code
 - TP01AppDelegate.h
 - TP01AppDelegate.m
 - TrotAppDelegate.ii
 - EAGLView.h



- Introduction
 - Objectifs du chaptire
- 2 Prérequis
 - Moteur de jeu
 - Xcode
 - Premier projet sous Xcode
 - Délégation et Protocol
 - Organisation du code
 - TP01AppDelegate.h
 - TP01AppDelegate.m
 - EAGLView.h



- définition d'un moteur de jeu
- introduction à Xcode
- premier projet sous xcode
- comprendre la délagation et le protocole
- lecture de code objectiveC

- Introduction
 - Objectifs du chaptire
- Prérequis
 - Moteur de jeu
 - Xcode
 - Premier projet sous Xcode
 - Délégation et Protocol
 - Organisation du code
 - TP01AppDelegate.h
 - TP01AppDelegate.m
 - EAGLView.h

Definition

Un moteur de jeu est un ensemble de briques/modules logiciels offrant des fonctionnalités génériques facilitant le développement d'un jeu vidéo.

Les fonctionnalités d'un moteur de jeu sont par exemple :

- Game Loop : gestion du cycle du jeu i.e comment réaliser les MAJ de la logique du jeu et du rendu graphique
- Rendu graphique : gestion du rendu des images + sprites + animations + fonts + tile maps + effets (particles)
- Son : gestion du son i.e comment lire et produire des effets sonores
- Inputs : gestion des entrées des utilisateurs
- GUI : gestion de l'interface (menus, widgets etc)
- Collision : gestion des collisions entre les entités de jeu
- Réseau : pour les jeux connectés



- Introduction
 - Objectifs du chaptire
- Prérequis
 - Moteur de jeu
 - Xcode
 - Premier projet sous Xcode
 - Délégation et Protocol
 - Organisation du code
 - TP01AppDelegate.h
 - TP01AppDelegate.m
 - EAGLView.h

Objectifs de ce chapitre :

- Se familiariser avec l'environnement xcode
- Créer son premier projet OpenGL SE

- Xcode est un IDE qui permet de programmer facilement sous Mac (pour iOS)
- On dispose d'un SDK iOS + Simulateurs (iphone, ipad)

- Vous pouvez créer votre premier projet en partant d'un template existant
- Nous allons par exemple créer un premier projet avec le template OpenGL SE
- Sélectionner IPhone comme produit et sauvegardez votre application par exemple avec le nom TP1
- Lancez l'application dans le simulateur avec le bouton Build & Run

Délégation

Le pattern de délégation

- Avec la délégation un objet délègue la réalisation d'une méthode à un autre objet.
- helloWorld { [monDelegue helloWorld] ; }

Délégation

Le pattern de délégation

- Avec la délégation un objet délègue la réalisation d'une méthode à un autre objet.
- helloWorld { [monDelegue helloWorld] ; }

Protocole dans objective C

Un protocole en objective-C est un ensemble de méthodes qu'une classe doit implémenter pour y adhérer

Organisation du code

- Nous séparons les header .h et les implémentations .m
- Le header contient
 - La déclaration des variables d'instances
 - Déclaration des méthodes d'instances (-) et de classe (+)
- Les propriétés exposées aux autres (@property)

TP01AppDelegate.h

```
#import < UIKit / UIKit . h>
@class EAGLView:
@interface TP01AppDelegate : NSObject <
       UIApplicationDelegate > {
    UIWindow *window:
    EAGLView *glView;
@property (nonatomic, retain) IBOutlet UIWindow *
       window:
@property (nonatomic, retain) IBOutlet EAGLView *
       glView:
@end
```

Protocol UIApplicationDelegate

Dans le protocole UIApplicationDelegate nous trouvons :

- didFinishLaunchingWithOptions : message envoyé quand l'application a été initialisée et lancée et juste avant de recevoir son premier événement application
- WillResignActive : message envoyé juste avant que l'application ne soit désactivée (bouton Home)
- applicationWillTerminate : envoyé juste avant que l'application se termine



Implémentation dans TP01AppDelegate.m

```
#import "TP01AppDelegate.h"
#import "EAGLView.h"
@implementation TP01AppDelegate
@synthesize window;
@synthesize glView;

    (BOOL) application: (UIApplication *) application didFinishLaunchingWithOptions: (NSDictionary

             *)launchOptions
    [glView startAnimation]:
    return YES:
  (void) application Will Resign Active : (UIApplication *) application
    [glView stopAnimation];
  (void) application Did Become Active : (UIApplication *) application
    [glView startAnimation];
  (void) application Will Terminate: (UIApplication *) application
    [alView stopAnimation]:
  (void) dealloc
                                                                     4 D > 4 B > 4 B > 4 B >
    [window release]:
```

@synthesis et @property

- @synthesis : est un tag qui permet de générer le code pour le getter/setter d'une propriété déclarée avec @property
- Consulter le fichier TP01AppDelegate.m

- @property(readonly, nonatomic, getter=isAnimating)
- readonly : lecture seule de la propriété (pas de setter)
- nonatomic : ne pas générer de synchronisation multi-thread
- getter : donner explicitement le nom du getter
- : déclare une méthode d'instance
- + : déclare une méthode de classe