Logiciels embarqués ambiants/iOS

Chapitre 2 : Game loop

Dr. Abdelkader Gouaïch[1]

[1]Department of Computer Science
Université de Montpellier

2012

Outline

Outline

Objetifs du chapitre

- Comprendre la notion de boucle de jeu
- Structure d'un jeu
- Introduction au classes et catégories en objective-C

La boucle de jeu

Les animations dans un jeu vidéo reposent sur un principe simple : L'affichage successif et rapide d'images pour créer une *illusion* de mouvement

Cycles R et U

La boucle de jeu spécifie comment sont organisées les différents cycles R et U

- Cycle d'affichage (R) : préparer l'imager à dessiner sur l'écran (render) et quand l'afficher effectivement
- Cycle de la logique de jeu (U) : mise à jour de la logique du jeu
  - inputs
  - collision
  - comportements, IA
  - musique

Mesurs de vitesse dans un jeu

On mesure dans un jeu deux vitesses : une liée au rendu graphique (R) et l'autre à la logique du jeu (U)

Frame Per Second (FPS)

- Frame Per Second (FPS) : est une unité de mesure qui donne le nombre de frame (images) dessinées par seconde
- iPhone par exemple offre un taux de rafraichissement de l'écran de 60 FPS

update speed

- Update Speed : fréquence avec laquelle la logique du jeu est mise à jour.
- Nous spécifions donc le nombre d'opérations de type U par seconde.

Outline

Solution 1 : une boucle simple

```
while(gameisRunning)
{
updateGameLogic(); //U
render(); //R
}
```

Analyse de la boucle simple

- La boucle simple introduit un problème de dépendance par rapport à la vitesse d'exécution de la plateforme cible
- Sur une machine puissante nous allons effectuer plus d'opérations U + R donc le jeu sera plus rapide
- Sur une machine moins puissante, nous allons effectuer moins d'opérations U+R, le jeu sera plus lent

Outline

Boucle Delta

- Déterminer un delta pour effectuer les opérations R+U
- Avantage : Le jeu va avoir la même vitesse indépendamment de la puissance de la machine
- Inconvénient : si le delta est grand on peut ignorer des événements comme par exemple les inputs.
- En effet, les inputs risquent de se cumuler/écraser avant de produire leurs effets

Outline

Time based fixed interval

- On effectue les opérations U avec un intervalle régulier
- Les opérations R sont effectuées en suivants le taux de rafraichissement de la carte graphique

Modification dans ES1Renderer

Catégorie Objective C

- Objective-C permet de créer des catégories
- Une catégorie permet de rajouter des méthodes à une classe de base ; c'est une alternative à l'héritage
- Les nouvelles méthodes sont accessibles à toutes les instances de la classe
- Dans ES1Renderer on déclare une catégorie Private avec une seule méthode : initOpenGL

Catégorie Objective C

- Objective-C permet de créer des catégories
- Une catégorie permet de rajouter des méthodes à une classe de base ; c'est une alternative à l'héritage
- Les nouvelles méthodes sont accessibles à toutes les instances de la classe
- Dans ES1Renderer on déclare une catégorie Private avec une seule méthode : initOpenGL

Outline

Initialisation de OpenGL

- Nous n'allons pas présenter dans le détail OpenGL ; nous allons simplement présenter les concepts nécessaires pour le cours.
- OpenGL offre une machine à état programmable pour dessiner des scènes (objets 3D)
-

Etape 1 : Matrice de projection

1ere étape configuration de la matrice de projection

- La matrice de projection permet de définir comment la scène est projetée (filmée)
- Orthogonale : une projection directe/orthogonale des points ; pas de distorsion
- Perspective : une projection avec une perspective ; distorsion des objets

2eme étape configuration de la matrice ModelView

- Cette transformation place les objets 3D et également la caméra virtuelle
- Nous devons passer explicitement dans le mode ModelView
- Mise en place d'une couleur background pour effacer le cadre
- Désactivation des tests de profondeur (simule diffusion de la lumière)
- Activation des fonctionnalités d'openGL, car par défaut les fonctionnalités sont désactivées.

La scène dans un jeu

- Le jeu est composé d'un ensemble de scènes
- Chaque scène va implémenter ses propres fonctons de rendu et de logique

classe GameController

- Cette classe donne un singleton qui est en charge du contrôle du jeu
- Nous allons utiliser la macro SYNTHESIZE_SINGLETON_FOR_CLASS(classname)
- Cette macro à consulter permet de construire un singleton pour une classe

TP02-Exo.zip

- Télécharger le fichier TP02-Exo.zip

classe GameScene

Pour le TP

- Consultez le fichier GameScene.h
- Consultez le fichier GameScene.m
- Consultez en particulier la fonction updateSceneWithDelta

Classe GameController

- Consultez le fichier GameController.h
- Consultez le fichier GameController.m
- Consultez le fichier SynthesizeSingleton.h

Objetif du TP

1. Créer une application qui fait monter et descendre le carré
2. Faire des rotations au carré
3. Tester l'application sur le simulateur