MOTEURS DE JEUX
INTRODUCTION

Marc Moulis, Septembre 2013

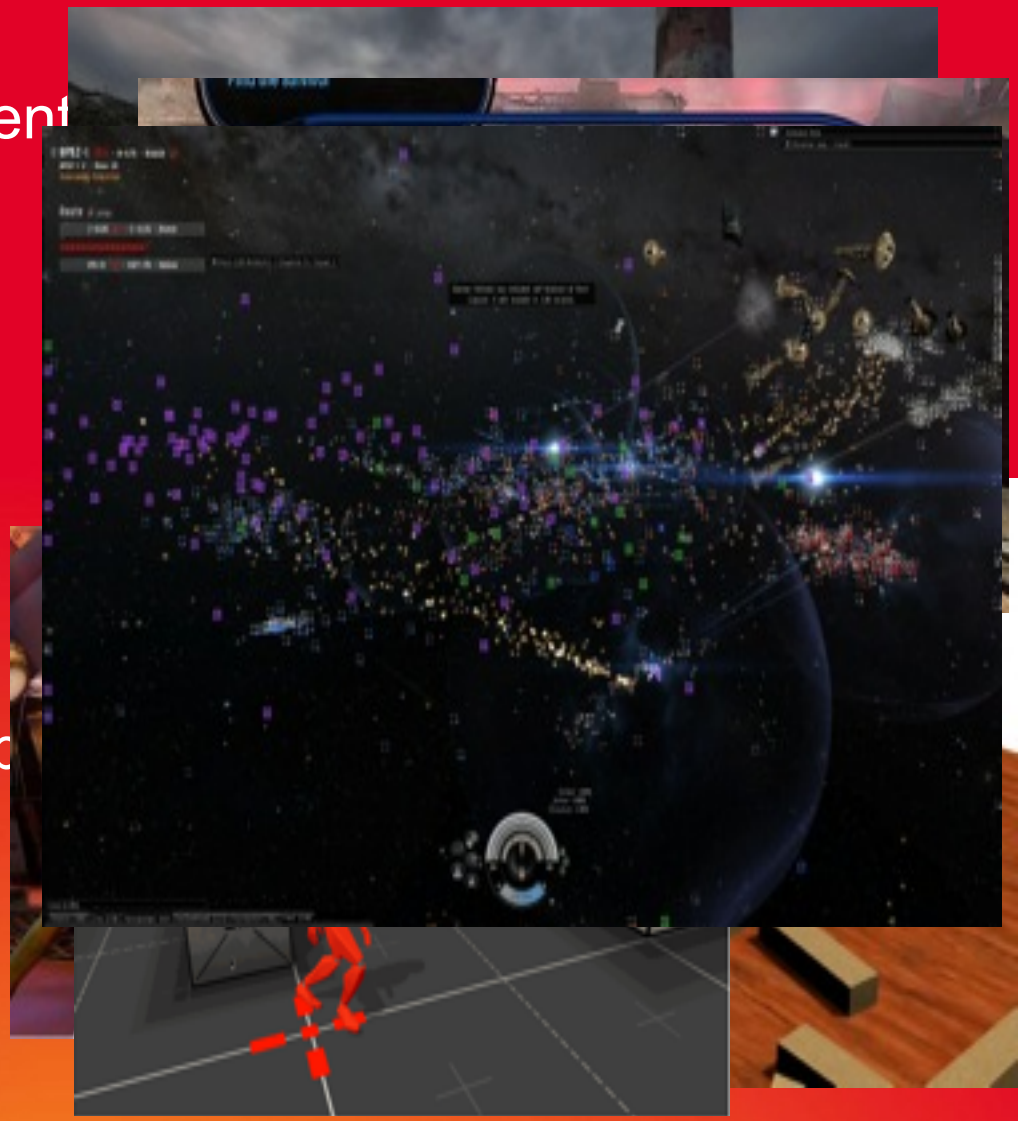
MOTEURS DE JEUX

1. Workflow
   - Corps de métiers
   - Processus de production

2. Structure d'un moteur de jeu
   - Définition
   - Problématique multi-plateformes

What is a 3D game engine?

- Core set of component that facilitate game creation
- Rendering
- Physics
- Sound
- User input
- Artificial intelligence
- Networking
- No such thing as an engine that can supp every type of game



Real-time strategy (RTS) engines

- Large number of low-detail game units
- Multiple levels of AI (individual units as well as computer players)
- Heightmap-based terrain
- Client/server

Minecraft (the real game)

- Procedurally-generated block world (voxel-based)
- Simple, pixelated graphics
- Undirected multiplayer gameplay
- Players manipulate the shape of the world

Starter code

Based on Qt framework

- Huge, cross-platform framework

Qt handles application loop

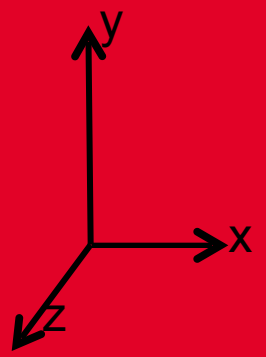
- Your code resides in callbacks
- See comments in method stubs for more details

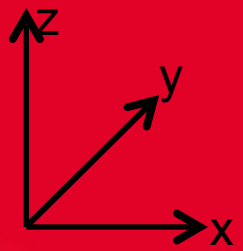
We strongly recommend separating your engine from QGLWidget (view.cpp)

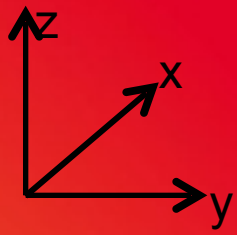
Inria

Coordinate systems

- Different game engines define coordinate systems differently
- Most of you will probably use the OpenGL coordinate system
- TAs will strive to be coordinate-system independent
- "Horizontal plane"
- Plane parallel to the ground (in OpenGL, the xz-plane)
- "Up-axis"
- Axis perpendicular to horizontal plane (in OpenGL, the y-axis)







OpenGL matrix transformations

OpenGL conflates them into two matrices (projection, modelview)

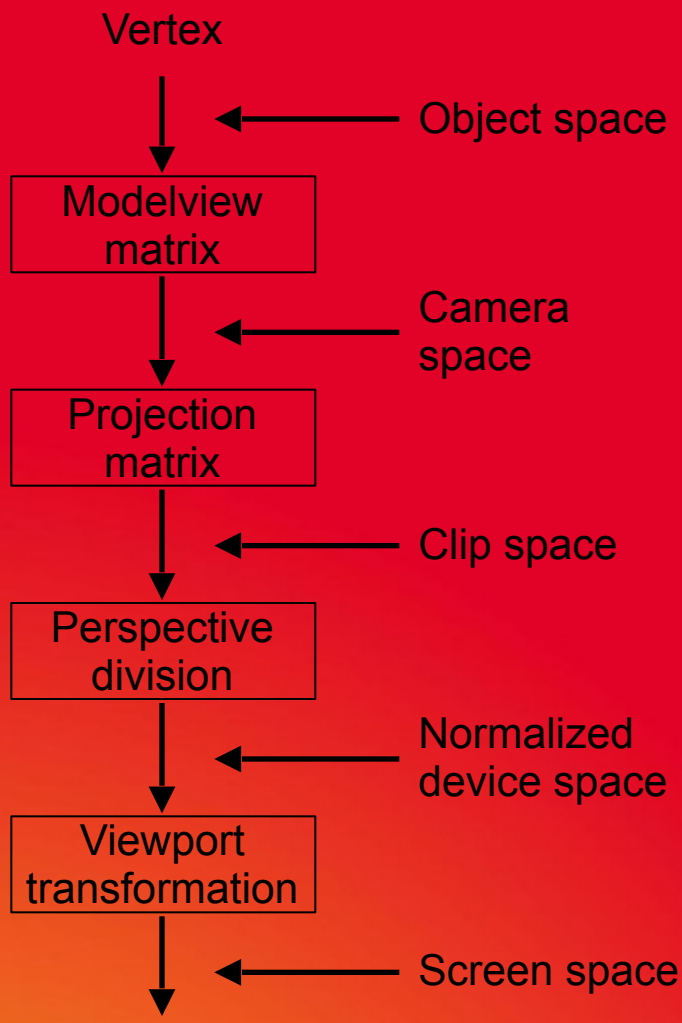
- Stored in column-major order rather than row-major

Modelview matrix

- Transforms object space to camera space, usually changes every frame

Projection matrix

- Has camera parameters (aspect ratio, field of view), usually only changes on window resize



WORKFLOW

WORKFLOW

ETAPES DE FABRICATION D'UN JEU

Avant d'étudier en détail les composants d'un moteur de jeu, il est important de se familiariser avec le processus de développement d'un jeu.

Selon l'ampleur et le type de projet, tous les corps de métiers ci-après peuvent ne pas être représentés, ou certaines personnes peuvent endosser plusieurs casquettes.

- Artistique
  - Directeur artistique
  - Concept artist
  - Graphiste (2D, 3D)
  - Graphiste technique
  - Animateur
  - Designer sonore
  - Musicien

- Technique
  - Directeur technique
  - Développeur

Production
  - Producteur

WORKFLOW

ETAPES DE FABRICATION D'UN JEU

La liste de ces étapes n'est pas forcément exhaustive, et peut varier d'un projet à l'autre ou d'un studio à l'autre. Néanmoins on retrouvera à chaque fois la même structure dans le déroulement de la production.

- Brainstorming
  - Propositions de concepts
- Pré-production
  - Ecriture du game design document (GDD)
  - Ecriture du technical design document (TDD)
  - Recherches artistiques
  - Mise en place de la chaîne d'outils (export, éditeurs, ...)

STRUCTURE DES MOTEURS DE JEUX

STRUCTURE DES MOTEURS DE JEUX

DÉFINITION

On appelle "moteur de jeu" l'ensemble des composants logiciels qui fournissent tous les services nécessaires à l'évolution et l'affichage d'un univers interactif, à vocation ludique.

On fera la distinction entre:

- Moteurs first-party: le moteur est tout ou majoritairement développé en interne par le développeur du jeu
- Moteurs third-party: le moteur est acquis auprès d'une société tierce qui l'a développé (Ex: Unreal Engine, Frostbite, Unity…)

STRUCTURE DES MOTEURS DE JEUX

DÉFINITION

On distingue globalement 2 courants:

- Moteurs généralistes:

  Les composants fournissent tous les services utiles pour la mise en œuvre de virtuellement n'importe quel type de jeu (c'est la tendance des moteurs third party : Renderware, Unreal Engine, Unity, Ogre, etc...)

- Moteurs dédiés:

  Les composants du moteur de jeu sont spécialisés pour un type de jeu précis: FPS, course, aventure, plateforme, RTS, etc...

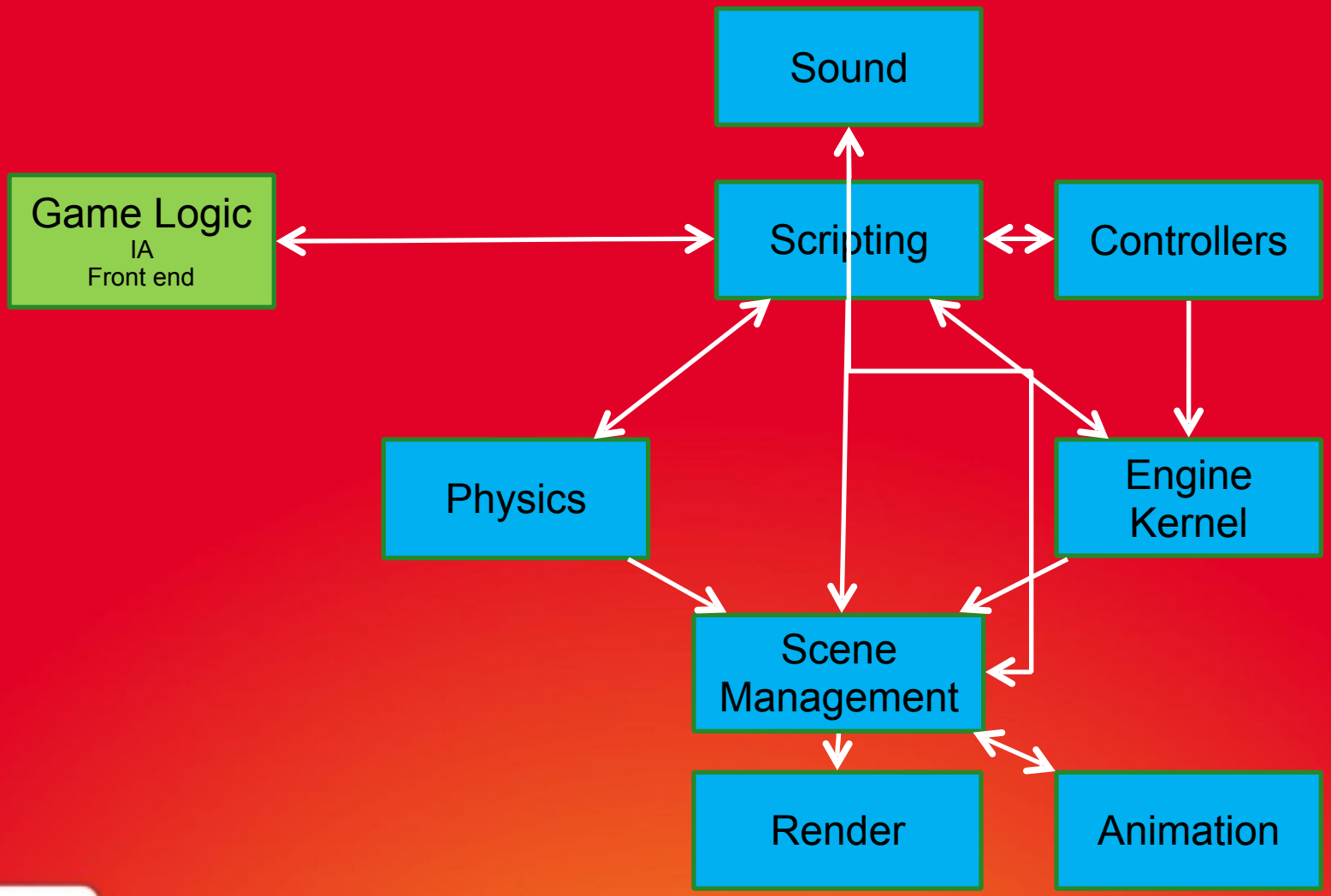
NB: *Dans le cadre de cette présentation, nous nous intéresserons principalement à la mise en place d'un framework qui pourrait servir de base commune tant à un moteur généraliste que dédié. Les spécificités techniques de chaque catégorie de jeu ne seront donc pas ou peu abordées.*

STRUCTURE DES MOTEURS DE JEUX

PROBLÉMATIQUES MULTI-PLATEFORMES

STRUCTURE DES MOTEURS DE JEUX

SCHÉMA CONCEPTUEL



STRUCTURE DES MOTEURS DE JEUX

PROBLÉMATIQUES MULTI-PLATEFORMES

Au vu des moyens techniques et humains nécessaires pour la production d'un jeu moderne, les développeurs font souvent le choix de publier le jeu sur plusieurs machines pour maximiser la rentabilité.

Le problème est que le développement sur chaque type de machine est différent:

- Organisation du code (ex: **multi-coeurs** vs **multi-processeurs** dédiés)
- Organisation et capacité de la mémoire (ex: **UMA** vs **mémoires dédiées**)
- Contraintes de publication (**TRC**)

STRUCTURE DES MOTEURS DE JEUX

PROBLÉMATIQUES MULTI-PLATEFORMES

Le challenge lors de l'écriture d'un moteur multi-plateformes est donc:

- De maximiser la mise en commun des composants logiciels d'une plateforme à l'autre
- De minimiser le nivelage par le bas

L'idée est de construire l'ensemble des composants logiciels (génériques) du moteur de jeu sur une base logicielle dédiée (donc spécifique) à chaque plateforme: ce qu'on appelle un couche d'abstraction (**HAL**).

Les avantages sont multiples:

- Le développement du moteur et du jeu deviennent (quasi-)indépendants de la plateforme cible
- Les développements des différents composants sont relativement décorrélés et donc parallèlisables

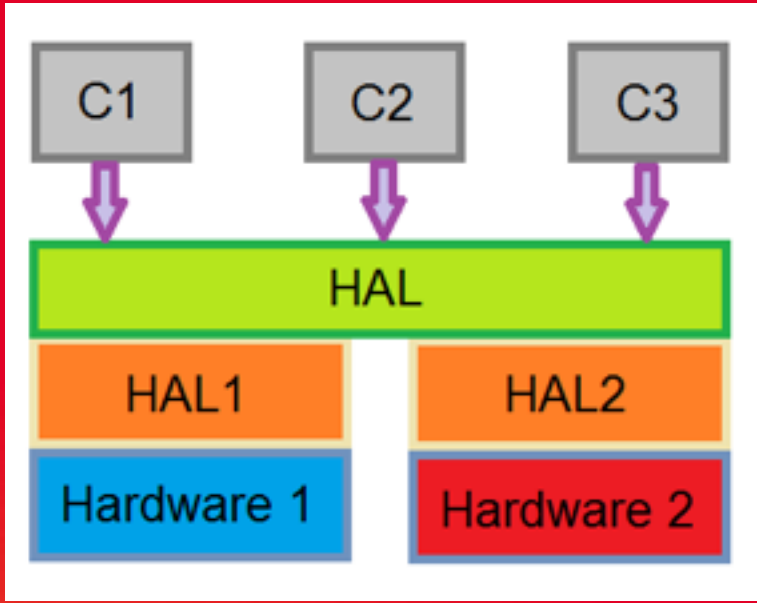
Inconvénient: comment éviter le nivelage par le bas ?

STRUCTURE DES MOTEURS DE JEUX

PROBLÉMATIQUES MULTI-PLATEFORMES

On pourra donc utiliser les stratégies suivantes:

- Redéfinition des types de données de base
- Surcharge de toutes les fonctions vitales (gestion mémoire, manipulation de chaînes, gestion des noms de fichiers, accès système)
- Mise en place d'une couche d'abstraction matérielle (I/O, rendu, multi-threading)
- Gestion d'un pool de ressources "dédiées" (ex: les icones représentant les boutons du pad)



Composants (C1, C2, C3) et couches d'abstraction matérielle (H1, H2)

Prochain cours : mathématiques 3D