3D temps réel avec Irrlicht

Facile Licence

INTRODUCTION

Commençons comme il est d'usage par un chapitre d'introduction qui va se contenter de présenter les tenants et aboutissants du sujet de ce tutoriel. J'ai nommé les **moteurs 3D**, et plus particulièrement Irrlicht.

Si vous savez déjà de quoi il s'agit ou que vous n'avez pas besoin de le savoir, lisez-le quand même. Juste pour être sûr...

Ce qu'est un moteur 3D

Question plus simple qu'il n'y parait. Pour faire court, un moteur 3D est une bibliothèque logicielle qui va s'occuper de tout ce qui concerne l'affichage à l'écran, les calculs dans l'espace, la lecture des fichiers externes, etc... dans une application 3D. D'ailleurs on parle aussi souvent de moteur graphique, l'aspect 3D n'est pas forcément le cœur. Ça dépend de quelle manière on voit les choses.

Maintenant que c'est dit, il reste quand même quelques points à éclaircir. Particulièrement pour les personnes qui ne sont pas très familières avec le monde de la programmation. Je rappelle ceci-dit qu'il faut avoir un certain niveau en C++ pour suivre ce tutoriel dans de bonnes conditions.

Intéressons nous de plus près au concept de **bibliothèque logicielle** (**library** en anglais). Il s'agit comme son nom l'indique d'une collection de fonctions logicielles regroupées dans un ou plusieurs fichiers. Le programmeur qui utilise une bibliothèque peut donc faire appel à ces fonctions dans les programmes qu'il crée. Par exemple avec Irrlicht, il suffit de faire appel à quelques fonctions pour créer une fenêtre et l'afficher à l'écran. Puis à quelques autres pour charger un personnage en 3D et l'afficher dans cette fenêtre, etc...

Il est important de faire la distinction entre les différents niveaux de bibliothèques logicielles.

En programmation le **niveau** désigne la proximité avec la machine. Plus un langage ou une bibliothèque est bas niveau et plus elle est proche du fonctionnement de la machine. Irrlicht, ou n'importe quel moteur 3D en général, est très haut niveau. Hormis les programmeurs bidouillant l'intérieur des moteurs, la plupart ne savent pas comment ça marche à l'intérieur et se contentent d'utiliser l'interface de programmation fournie telle quel.

Cette interface justement porte un nom qui va revenir très régulièrement dans ce tutoriel : **API**, pour **Application Programming Interface**. Comme son nom l'indique une API est l'interface par laquelle le programmeur va "communiquer" avec la bibliothèque. Elle est constituée de toutes les fonctions "publiques" librement accessibles.

Par abus de langage on utilise parfois l'acronyme API pour désigner la bibliothèque à laquelle elle permet d'accéder.

Il existe des APIs très bas niveau qui accèdent directement ou presque aux éléments **hardware**, aux éléments physiques de la machine. En les utilisant comme interface on peut donc accéder indirectement à ces éléments hardware. Ces manipulations sont particulièrement délicates dans le cas d'un moteur portable (i.e. qui fonctionne sur les principaux systèmes d'exploitation), car il faut alors adapter certaines parties du code spécifiques à chaque OS.

Le fenêtrage et la gestion de périphériques est spécifique à chaque OS par exemple. Sous Windows il faut utiliser la WinAPI alors que sous Linux on utilise généralement X Window.

Si vous n'avez pas compris la moitié de ce que vous venez de lire, ce n'est pas fondamentalement gênant puisque Irrlicht gère tout cela de manière transparente.

Voici un petit graphique récapitulatif de tout ce que nous venons de voir par niveaux :

Moteur 3D

APIs bas niveau : OpenGL, Direct3D, X, etc...

OS (système d'exploitation)

Hardware (matériel)

Irrlicht

Rentrons un peu plus dans le vif du sujet avec une présentation du moteur qui nous intéresse : **Irrlicht**. Nous n'allons pas tout de suite voir comment l'utiliser mais plutôt commencer par le décrire. Allez hop, trois petits screens au passage pour vous faire envie (d'autres sont disponibles dans la galerie sur le site officiel [21-1]) :

(Cliquer pour agrandir)

Un peu de culture

Le projet Irrlicht débute en 2002 en Autriche. C'est **Nikolaus Gebhardt** (en photo à droite), programmeur de métier, qui eut l'idée de se lancer dans la réalisation d'un moteur 3D. Depuis le moteur en est à la version 1.7 et de très nombreuses améliorations ont été apportées, dont on peut voir la liste sur le site officiel [21-1].

L'équipe s'est également agrandie puisqu'il y a actuellement heu... trop de membres pour que je les compte. Mais la liste en est également visible sur le site. Et pour finir une anecdote à ressortir pour briller en société : irrlicht signifie "feu follet" en allemand.

Un peu de technique

Maintenant que nous savons un peu mieux ce qu'est un moteur 3D, une question doit sûrement vous brûler les lèvres :

Pourquoi existe-t-il plusieurs moteurs 3D s'ils font tous la même chose ? Et pourquoi choisir Irrlicht plutôt qu'un autre ?

Tout d'abord il faut bien se rendre compte que tous les moteurs 3D **ne** font **pas** la même chose. Ils ont la même fonction, mais certains sont plus performants ou complets que d'autres. Cela doit d'ailleurs vous paraître une évidence si vous jouez régulièrement à des jeux vidéos actuels. Quant au choix d'Irrlicht plutôt qu'un autre moteur, voici quelques arguments intéressants :

- **Il est libre et multi-plate-formes.** Ce qui signifie que le code source est librement téléchargeable, utilisable comme bon vous semble dans le respect de la licence, et fonctionne sous tous les OS ou presque.
- **Il est codé en C++ et complètement orienté objet.** Ce qui lui permet d'être plutôt intuitif et très bien organisé.
- **Il gère nativement beaucoup de formats de fichiers.** Que ce soit fichiers images , modèles 3D ou même archives [21-6]. Ce qui évite d'avoir recours à des *loaders* externes.
- **Il gère nativement plusieurs effets intéressants.** Comme des systèmes de particules, génération de terrains via heightmap, etc... Ce qui évite d'avoir à les coder soi-même.
- **Il est relativement simple à mettre en place et à maîtriser.** Grâce notamment à sa très bonne documentation et sa communauté large et active.

Irrlicht et Ogre 3D

La plupart du temps, quand quelqu'un a envie de coder une application 3D en C++ sans réinventer la roue, il en arrive souvent à ce dilemme cornélien : **Irrlicht ou Ogre 3D**[21-4]. Pour avoir utilisé les deux en profondeur, la principale différence se situe dans l'opposition puissance/simplicité. Irrlicht est de loin plus simple qu'Ogre, mais en contrepartie Ogre est de loin plus puissant qu'Irrlicht.
Par exemple Irrlicht est relativement monolithique, le moteur reste complet peu importe ce qui sera utilisé au final dans l'application, ce qui permet une très grande simplicité dans la mise en place d'un projet. D'un autre coté Ogre est beaucoup plus modulaire, utilisant un système de plugins qui permet de n'utiliser que ce dont on a besoin, mais qui est du coup plus complexe à mettre en place.
Si vous avez un doute sur le moteur que vous voulez utiliser ou que vous êtes juste curieux, le meilleur moyen de se faire une idée est de tester les deux. Il existe justement un tutoriel concernant Ogre 3D sur ce site même [21-5], n'hésitez pas à y jeter un œil. Contrairement à ce que laissent parfois penser les échanges sur les forums, utiliser un moteur ne condamne pas à rejeter les autres, il n'y a pas de dogme à adopter. Prenez le temps nécessaire pour "sentir" celui qui vous plait le plus ou qui correspond le mieux à vos besoins avant de faire un choix (qui ne sera pas définitif de toute manière).

3D temps réel avec Irrlicht

Mise en place

L'auteur

Kevin Bockelandt

Découvrez aussi ce cours en...

PDF