Programmation OpenGL/C
JUMPING BANANA

Samuel Jacquet

07 juillet 2019

Table des matières

1 Structure du projet

```
/repository
├── release
│   ├── file
│   ├── pics
│   ├── freeglut.dll
│   ├── DevIL.dll
│   ├── DevIL.lib
│   ├── ILU.dll
│   ├── ILU.lib
│   ├── ILUT.dll
│   └── ILUT.lib
├── build
├── include
│   ├── GL
│   │   ├── freeglut.h
│   │   ├── freeglut_ext.h
│   │   ├── freeglut_std.h
│   │   ├── gl.h
│   │   ├── glu.h
│   │   ├── glut.h
│   │   └── glext.h
│   ├── IL
│   │   ├── il.h
│   │   ├── ilu.h
│   │   └── ilut.h
│   ├── map.h
│   ├── game.h
│   ├── char.h
│   ├── score.h
│   └── tm.h
├── lib
│   ├── libfreeglut.a
│   └── libfreeglut_static.a
├── report
│   ├── pic
│   ├── report.pdf
│   ├── report.gz
│   ├── report.tex
│   └── references.bib
└── src
    └── map.c
```

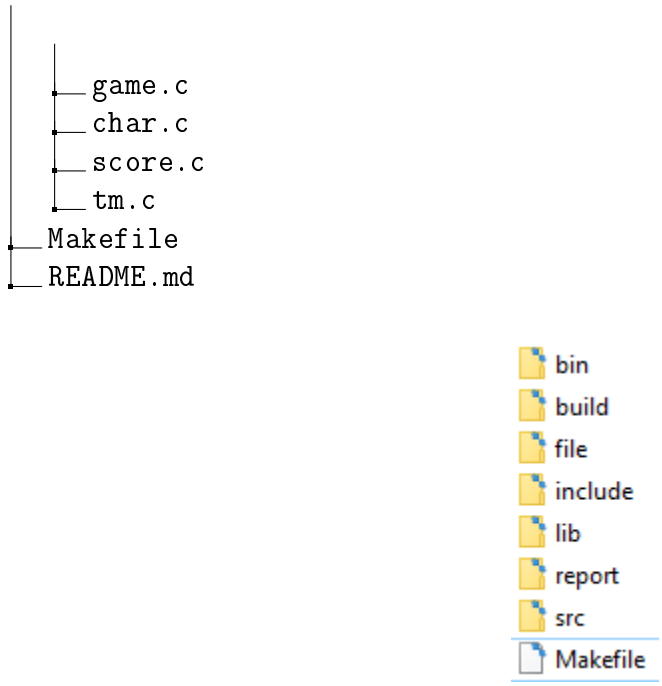

FIGURE 1 – Ma structure

— .h : header file, its a source file containing declarations (as opposed to .cpp, .cxx, etc. containing implementations),

— .lib : static library may contain code or just links to a dynamic library. Either way it's compiled code that you link with your program. The static library is included in your .exe at link time.

— .dll : dynamic library. Just like a static one but you need to deploy it with your .exe file because it's loaded at run time.

Il y a effectivement des difference dans tout ces types d'extension... - Le .a est lié à la compilation - Le .dll est lié à l'exécution

.a = statique (linux)(GCC clang)

.so = dynamique (linux)(GCC clang)

.lib = c'est perché (c'est windows), Librairie statique (Minimalist GNU ou visual Studio) ou d'importation (Visual Studio : https ://msdn.microsoft.com/fr-fr/library/kkt2hd12.aspx). A noté il me semble que les librairie statique minimalist GNU ne sont pas compatible avec visual studio la réciproque est fausse, excepté pour les librairie d'importation...

.dll = lib dynamique (Minimalist GNU, Visual Studio). Il me semble qu'il est possible de compiler des .so aussi (Minimalist GNU).

2 Enivronnement de compilation

3 Tile Mapping

This is a sample : [10].
This is a second line [7].
Le système GPS[1] *a priori*

3.1 Introduction

The motivation.

3.2 Afficher un tableau à deux dimensions

Parfois, il est utile de définir un tableau de pointeurs plutôt qu'un tableau en deux dimensions. L'argument formel d'une fonction prenant un tableau en deux dimensions en argument doit spécifier au moins la deuxième dimension.

```
int multiply(int x, int a[][10])
```

Nous sommes obligé de lui donner la deuxième dimension pour connaître la taille des lignes.

```
double Q[6][6] =
{
        {1,1,1,9,1,1},
        {1,1,1,1,1,1},
        {1,2,1,1,1,1},
        {1,1,1,1,1,1},
        {1,1,1,1,1,1},
        {1,1,1,1,1,1},
};
```

Un tableau à une dimension est un pointeur mais il ne faut pas confondre pointeur et tableau. Si on peut remplacer des crochets par une étoile, cela n'est vrai que pour la première dimension d'un tableau. Ainsi *char tab[10]* peut se voir comme un char *tab mais un *char tab[10][20]* ne pourra à la limite être vu que comme un *char (*tab)[20] ou tab[][6].*

```
void affiche3(double tab[][6])
void affiche2(double tab[6][6])
void affiche1(double (*tab)[6]) //un tab de 6 pointeurs vers des valeurs de
    types double.
// représente une matrice de 10 lignes ou le nombre de colonnes varient pour
    chaque ligne.
{
    int i, j;
    for(i=0; i<6; i++)
```

1. Global Positioning System

```
    {
        for(j=0; j<6; j++)
        {
            printf("%.f", tab[i][j]);
        }
        printf("\n");
    }
    printf("\n");
}
```

En mémoire, tout tableau est toujours converti en tableau à 1 dimension (une suite de cases) mais le compilateur a besoin de connaitre les *mesures* du tableau pour savoir comment calculer à quelle case correspond *tab[x][y]*. Or on peut oublier la première mesure, il lui faut quand-même les autres pour savoir de combien décaler. C'est comme pour un cube. Même avec son volume il faut quand-même 2 autres mesures pour trouver le 3 degré.

3.2.1 Les bonnes pratiques

```
#include <stdio.h>

int main(void) {
  printf("Hello World\n");
  return EXIT_SUCCESS;
}
```

4 Order lists and other lists

4.1 Linux building environment

- sudo apt-get update — Update the package index
- sudo apt-get install build-essential — Install essential like Makefile tools
- sudo apt-get install libdevil-dev — Install libdevil-dev deb package

4.2 Windows building environment

4.2.1 MINGW - Minimalist GNU for Windows

La mise en place de MinGW[5] va permettre d'éditer, compiler et démarrer le programme. MinGW est la contraction de *Minimalist GNU for Windows*, c'est un environnement de développement pour le compilateur GCC [2]. MinGW se réfère à un ensemble d'en-têtes d'exécution, utilisés dans la construction d'un système de compilation basé sur le GCC GNU et les projets binutils. Il compile et lie le code à exécuter sur les plates-formes Win32[5].
MinGW est disponible sur le site de sourceforge. Il faut également installer un éditeur de code comme par exemple notepadPlusPlus.
Il faut aussi indiquer à Windows où se trouve les fichiers utiles à la compilation et à l'éxécution (les bibliothèques dynamiques DLL [3] dans le dossier `C:\MinGW\bin` par exemple), c'est à dire ajouter un chemin dans les variables d'environnements sur Windows (dans les versions précédentes à Windows 10, il faut bien séparer les différents PATH par des points virgules)[3].

4.2.2 GCC - GNU Compiler Collection

GCC est le compilateur C de GNU.

GNU's Not Unix (GNU N'est pas Unix) est un système d'exploitation libre créé en 1983 par Richard Stallman. Il reprend les concepts et le fonctionnement d'UNIX. Parmi la liste des paquets GNU, on retrouve la collection de compilateurs GNU, le débogueur GDB ou encore le shell Bash.

4.2.3 GNU - Makefile

Le programme setup est disponible sur le site de sourceforge. A l'heure d'aujourd'hui, le dernier éxécutable est make-3.81.exe.

Pour l'installation sur Windows, il faut définir le chemin vers le répertoire des binaires dans les propriétés système, puis modifier la variable d'environnement et en selectionnant le

2. GNU Compiler Collection
3. Dynamic Link Library

chemin `PATH:C:\ProgramFiles\...;`.

Si ce n'est pas déjà fait, il faut s'assurer de copier le fichier make.exe dans `c:\MinGW\bin\` (`c:\MinGW\bin\make.exe`).

Pour une utilisation plus généralisé au travers du système Windows, nous pouvons également installé la bibliotheque dynamique DLL dans le répertoire `C:\Windows\System32` et dans le répertoire `C:\Windows\SysWOW64\` dans le cas d'une version 64-bits de Windows 10 (8,7,Vista et XP). Dans le cadre d'une simple utilisation pour un jeu vidéo 2D, ce n'est pas une obligation.

4.2.4 GNU - Git bash

GNU est principalement utilisé sur les systèmes d'exploitations de la famille des systèmes UNIX. Un moyen pour exécuter des lignes de commandes dans le style UNIX est d'utiliser une interface de type bash (ligne de commande shell) qui permet d'utiliser l'utilitaire Make dans l'environnement Windows.

Note : Pour modifier, afficher les extensions des fichiers ou afficher les fichiers cachés sur Windows 10, il faut ouvrir le Panneau de configuration, les options de l'Explorateur de fichiers, l'onglet affichage, décocher « Masquer les extensions des fichiers dont le type est connu »et cocher « Afficher les fichiers, dossiers ou lecteurs cachés ». Ce dernier permet d'observer l"initialisation d'un nouveau dépot (.git).

4.2.5 Graphics librairies - OpenGL

GLUT, OpenGL Utility Toolkit, étais écrit à l'origine par Mark Kilgard et porté vers Win32 (Windows 95,98,Me,NT,2000,XP) par Nate Robins. La dernière version de la bibliothèque est 3.7.6 (*Novembre, 2001*). Une alternative est la bibliothèque freeglut version 3.0.0. (*version stable, sorti en 2015*).

Télécharger et extraire les fichiers présent dans l'arborescence ci-dessous, à partir des sites officiels, avec un logiciel d'archives qui prend en charge les formats zip ou gz. Dans le cadre du projet de fin d'année en première année d'informatique à UNamur, il faut prélever le contenu des dossiers X86 pour utiliser freeglut dans sa version 32 bits pour éviter les problème de compilation.[2]

```
/freeglut
├── bin
│   └── freeglut.dll
└── include
```

```
├── GL
│   ├── freeglut.h
│   ├── freeglut_ext.h
│   ├── freeglut_std.h
│   ├── gl.h
│   ├── glu.h
│   ├── glut.h
│   └── glext.h
└── lib
    ├── libfreeglut.a
    └── libfreeglut_static.a
```

La plupart du temps, le téléchargement inclut un dossier *bin*, un dossier *include* et un dossier *lib* que l'ont peux intégrer à notre dossier de projet.

Dans le dossier bin, on retrouve une bibliothèque dynamique (freeglut.dll) qui est essentiel au bon fonctionnement de notre programme. Ce dossier contient également l'exécutable du projet. A la différence d'une bibliothèque statique, c'est que le fichier dll est lancé au moment de l'exécution du programme. Les bibliothèques statiques sont intégrées dans l'exécutable au moment de la compilation.

Le dossier include contient tous les fichiers d'en-têtes et le dossier lib contient des fichiers compilés, des fichiers objets.

A coté de ces dossiers, dans notre dossier projet, on peut ajouter un dossier *src* qui contient les fichiers c.

Dans nos fichiers .c, on peut intégrer *#include <GL/freeglut.h>*

Pour compiler, on démarre par le choix de notre compilateur *GCC*. Si on a choisit de compiler à partir de notre dossier parent vers un sous dossier *src*, on doit utiliser la commande `src/*.c` auquel le tiret grand i (*-I include/GL*) permet d'inclure les bibliothèques externes précédé du répertoire `include/GL`. Tandis que l'argument tiret grand L (*-L lib*) permet d'inclure les fichiers objets qui sont déjà compilés. Finalement l'arguement *-o* ***bin/prog*** pour insérer notre exécutable dans notre fichier binaire.

La compilation est accompagné d'options de liens spécifiques aux bibliothèques Glut et freeglut : *-lfreeglut -lopengl32 -lglu32*.

Les fichier OpenGL sont disponible sur le site de sourceforge ou encore mieux, en suivant les commandes shell de MinGW après l'installation de MSYS en suivant les instructions sur le site MinGW[6].

L'intérêt de cette dernière méthode est la possibilté de partager le dossier des binaires avec une tierce personne qui n'aura pas besoin d'installer OpenGL dans son ordinateur sous système d'exploitation Windows.

```
/freeglut
└── bin
    ├── freeglut.dll
    └── prog.exe
```

Autrement après l'installation de MinGW, il faut installer les composants de la manière suivante :

- Les bibliothèques statiques (*libfreeglut.a* et *libfreeglut_static.a*) dans le chemin `C:\MinGW\lib`.
- Les fichiers d'en-têtes dans le répertoire `C:\MinGW\include\GL`.
- les bibliothèques de liens dynamiques (*freeglut.dll*) sont installé dans le chemin des binaire `C:\MinGW\bin`.

4.2.6 Graphics librairies - DevIL

DevIL est une bibliothèque multiplateforme de gestion des images lancé par Denton Woods. Elle est capable de lire une grande quantité de formats d'image (en lecture et en écriture). DevIL peux charger les fichiers suivant .bmp, .cut, .dds, .doom, .exr, .hdr, .gif, .ico, .jp2, .jpg,.lbm, .mdl, .mng, .pal, .pbm, .pcd, .pcx, .pgm, .pic, .png, .ppm, .psd, .psp, .raw, .sgi, .tga et .tif. Pour installer cette bibliothèque, il faut se rendre sur le site Officiel de Sourceforge[4].

```
/DevIL Windows SDK
├── include
│   └── IL
│       ├── il.h
│       ├── ilu.h
│       └── ilut.h
└── lib
    └── x86
        └── Release
            ├── DevIL.dll
            ├── DevIL.lib
            ├── ILU.dll
            ├── ILU.lib
            ├── ILUT.dll
            └── ILUT.lib
```

4.2.7 Sounds librairies - (old) FMOD

Fmod est une bibliothèque multiplateforme de gestion du son (MP3[4], RAW[5], FLAC[6], WAV[7], WMA[8],ect). On peux trouver ces libraries sur le site de Sourceforge[11].

4.3 Some errors that occured

4.3.1 OpenGL

Entre glBegin et glEnd, il existe qu'une série de fonction qui sont acceptées : glVertex, glColor, glIndex, glSecondaryColor, glNormal, glMaterial, glFogCoord, glTexCoord, glMultiTexCoord, glEgleFlag, glArrayElement, glEvalCoord, glEvalPoint, glVertexAttrib, glUniform, et c'est tout ! Toutes les autres opérations ne fonctionneront pas[1].

4.3.2 linking error

Lorsque j'ai commencé à inclure mes bibliothèques, notament de gestion du son, j'ai rencontré des problèmes à la compilation. J'ai souvent perdu de nombreuses heures à trouver la bonne option vers mes librairies pré-compilées.

```
build/audio.o: In function `comp':
C:\Users\jacqu\Desktop\jeu\1819_IHDCB132_Jacquet_Samuel_2ndSession/src/audio.c
    :25: undefined reference to `FMOD_System_Create@4'
```

```
c:/mingw/bin/../lib/gcc/mingw32/6.3.0/../../../../mingw32/bin/ld.exe: cannot
    find -lfmod
```

```
gcc: warning: lib\fmod_vc.lib: linker input file unused because linking not
    done
```

```
c:/mingw/bin/../lib/gcc/mingw32/6.3.0/../../../../mingw32/bin/ld.exe: cannot
    find include/FMOD/fmod.h
```

4. MPEG I/II couche 3
5. Global Positioning System
6. Free Lossless Audio Codec
7. Waveform Audio File Format. Edité par Microsoft et IBM.
8. Windows Media Audio format. Egalement édite par Microsoft.

Pourtant la solution est simple, il faut indiquer à GCC le chemin vers ses libraries avec un l majuscule *-L lib* et ensuite je prend le nom de ma librairie précompilée sans son extension .a ou .lib (.so sous Linux) pour l'ajouter à ma commande de compilation avec un l minuscule *-lfmod_ vc*.

4.4 GitHub commands

1. Getting and Creating Projects
   - git init — Initialize a local Git repository.
   - git clone `https://github.com/jacquets/1819_IHDCB132_Jacquet_Samuel_2ndSession.git`
     Create a local copy of a remote repository.
2. Base
   - git status — Check status.
   - git add — Add a file to the staging area.
   - git commit -m « [commit message] » — Commit changes.
   - git rm -r [file-name] — Remove a file (or folder).
   - Sharing and Updating Projects
   - git push origin [branch name] — Push a branch to your remote repository.
   - git push -u origin [branch name] — Push changes to remote repository (and remember the branch).
   - git push — Push changes to remote repository (remembered branch).
   - git push origin –delete [branch name] — Delete a remote branch.
   - git pull — Update local repository to the newest commit.
   - git pull origin [branch name] — Pull changes from remote repository.
   - git remote add origin `https://github.com/jacquets/1819_IHDCB132_Jacquet_Samuel_2ndSession.git` — Add a remote repository.
3. Branching and Merging
   - git branch — List branches (the asterisk denotes the current branch).
   - git branch -a — List all branches (local and remote).
   - git branch [branch name] — Create a new branch.
   - git branch -d [branch name] — Delete a branch.
   - git push origin –delete [branch name] — Delete a remote branch.
   - git checkout -b [branch name] — Create a new branch and switch to it.
   - git checkout -b [branch name] origin/[branch name] — Clone a remote branch and switch to it.

- git checkout [branch name] — Switch to a branch.
- git checkout – [file-name.txt] — Discard changes to a file.
- git merge [branch name] — Merge a branch into the active branch.
- git merge [source branch] [target branch] — Merge a branch into a target branch.
- git stash — Stash changes in a dirty working directory.
- git stash clear — Remove all stashed entries.

4. Inspection and Comparison
   - git log — View changes.
   - git log –summary — View changes (detailed).
   - git diff [source branch] [target branch] — Preview changes before merging.
   - git config –edit –global

4.5 Linux commands

4.5.1 sudo commands used

sudo -i	root mode.
su	super user mode.
sudo apt-get install build-essential	the main setting up. Installed with the make tool.
sudo apt update	used to update before instalation of a new packag
sudo apt-get update	
apt install gcc-mingw-w64-x86-64	sudo apt install vim
sudo apt install renamel	
sudo apt-get install imagemagick[9]	picture processing commands to change image for
sudo snap install atom –classicl	
sudo apt install synapticl	
sudo apt-get install libpng-dev	
sudo apt-get install zlib1g-dev	
apt-get install freeglut3 freeglut3-dev libglew-dev	to install Freeglut, it's the most up to date. sudo

4.6 Functions used list

4.6.1 GL

glColor3d(GLdouble red, GLdouble green, GLdouble blue);

4.6.2 IL

Références

[1] OpenClassRooms. https://openclassrooms.com/fr/, 2018.

[2] Jason Champagne. Langage C #23 - introduction SDLh. https://www.youtube.com/watch?v=Lwx9rSgwoDg&t=173s, 2017.

[3] Jason Champagne. Tutoriel Windows - variable d'environnement Path. https://www.youtube.com/watch?v=M2BWTJXDJXY&index=2&list=PLrSOXFDHBtfFrcRVrJ2ELX2_160l_CpQd, 2018.

[4] Meloni Dario et al. Denton Woods, Nicolas Weber. DevIL - Developer's Image Library. http://openil.sourceforge.net/, 2017 - last release.

[5] earnie. MinGW. http://www.mingw.org/wiki/mingw/, 2008.

[6] joshuaburkholder. HOWTO_Compile_the_OpenGL_Utility_Toolkit_GLUT_for_Win32_with_M http://www.mingw.org/wiki/HOWTO_Compile_the_OpenGL_Utility_Toolkit_GLUT_for_Win32_with_MinGW, 2010.

[7] Ingo Lütkebohle. BWorld Robot Control Software. https://tex.stackexchange.com/questions/281868/import-bib-file-into-tex-file, 2008. [Online ; accessed 19-July-2008].

[8] Lubos Rendek. How to install ImageMagick 7 on Ubuntu 18.04 Linux. https://linuxconfig.org/how-to-install-imagemagick-7-on-ubuntu-18-04-linux, 2018.

[9] ronan. Imagemagick. https://www.quennec.fr/trucs-astuces/syst%C3%A8mes/gnulinux/commandes/multim%C3%A9dia/photo/imagemagick, 2018.

[10] Smith, J. *Once Upon a Time on Stack Overflow*. United Publishers, Houston, TX, 1th edition, 2015.

[11] zachdavis. CaveWriting : 3D Hypermedia Authoring. https://sourceforge.net/p/cavewriting/code/HEAD/tree/CW3/, 2012.

samuel.jacquet@student.unamur.be