

计算机图形学理论和应用

OpenGL 实验环境准备

September 9, 2022

1 实验要求

- 完成 OpenGL 开发环境配置，为后续实验提供支持。
- 成功运行提供的[HelloWindow.cpp](#)文件则实验通过。
- 可选 Windows 或 Linux 等系统安装。
- 可选择使用其他 IDE 进行环境配置，如 Visual Studio。

2 WINDOWS 下 VSCode 配置 OPENGL 开发环境-快速版

如果想快速配置 OpenGL，可使用已下载好的文件进行配置，按以下流程操作即可；如果想自主配置，可阅读下一节。

1. 在 CMD 中输入 `mingw32-make -v`，如果报错，下载[MinGW](#)，解压至 C 盘根目录 (解压后为 C:MinGW)。
2. 上一步未报错可跳过此步，将 MinGW 安装位置"C:\MinGW\bin"添加到环境变量(注：根据安装位置修改路径，也可能解压出来是"C:\MinGW\MinGW\bin")。
3. 在 CMD 中输入 `mingw32-make -v`，如果输出信息有 i686 下载[此实验项目](#)，如果输出信息包含 x86_64 则下载[此项目](#)，解压。
4. 已安装 VSCode 跳过，下载安装[VSCode](#)。
5. 安装 C\C++ 和 C\C++ Project Genetator 插件。
6. 解压下载的项目，右键 CG_Lab_32 或 CG_Lan_64 通过 Code 打开。
7. 终端运行 `make run dir=hello_window`，运行成功即测试通过。
8. 测试结果见图 [3.8](#)。

3 WINDOWS 下 VSCode 配置 OPENGL 开发环境

3.1 安装 gcc、g++、gdb 和 mingw32-make(或 make)

3.1.1 验证是否已安装

在 CMD 中依次输入 `gcc -v`、`g++ -v`、`gdb -v` 和 `mingw32-make -v`(或 `make -v`)，如果输出版本信息，则说明已安装，可跳过此步，见图 3.1，如果输出错误信息，需配置上述环境，见图 3.2。

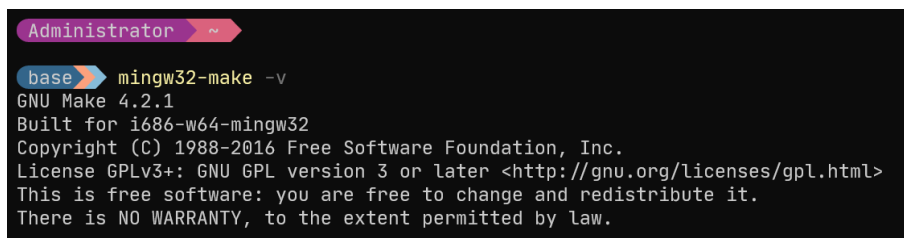


Figure 3.1: 已安装

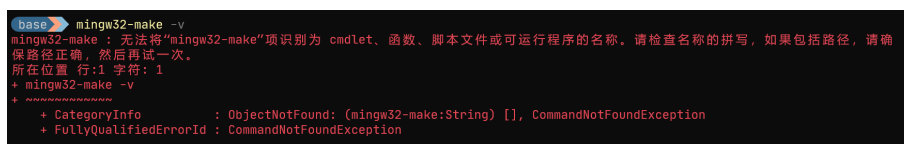


Figure 3.2: 未安装

3.1.2 下载安装 [MinGW](#), 或直接使用助教提供的 [MinGW\(32 位\) 压缩包](#)，解压使用，建议安装目录 [C:\MinGW](#)

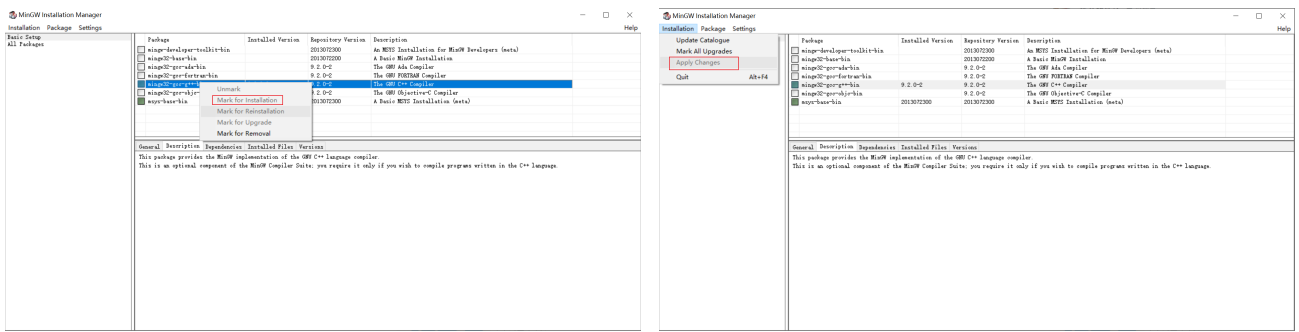
1. 下载安装成功后打开 MinGW Installation Manager。
2. 在安装界面 Basic Setup 找到 mingw32-gcc-g++-bin、msys-base-bin，在 All Packages 中找到 mingw-gdb-bin、mingw32-make-bin，点击选中后右键选择 Mark for Installation；4 个都选择后，点击界面左上角 Installation 选择展开的 Apply Changes，见图 3.3。(注：助教已经安装过了，所以右键界面会有部分不同)

3.1.3 配置环境变量

1. 将 MinGW 安装位置 "C:\MinGW\bin" 添加到环境变量(注：根据安装位置修改路径)。
2. 验证 gcc 等是否安装成功。

3.2 下载安装 VSCode

1. 安装完成后，在 VSCode 扩展一栏安装 C\C++ 和 C\C++ Project Genetator 插件。
2. 新建项目文件夹，如 "CG_Lab"，右键通过 Code 打开。

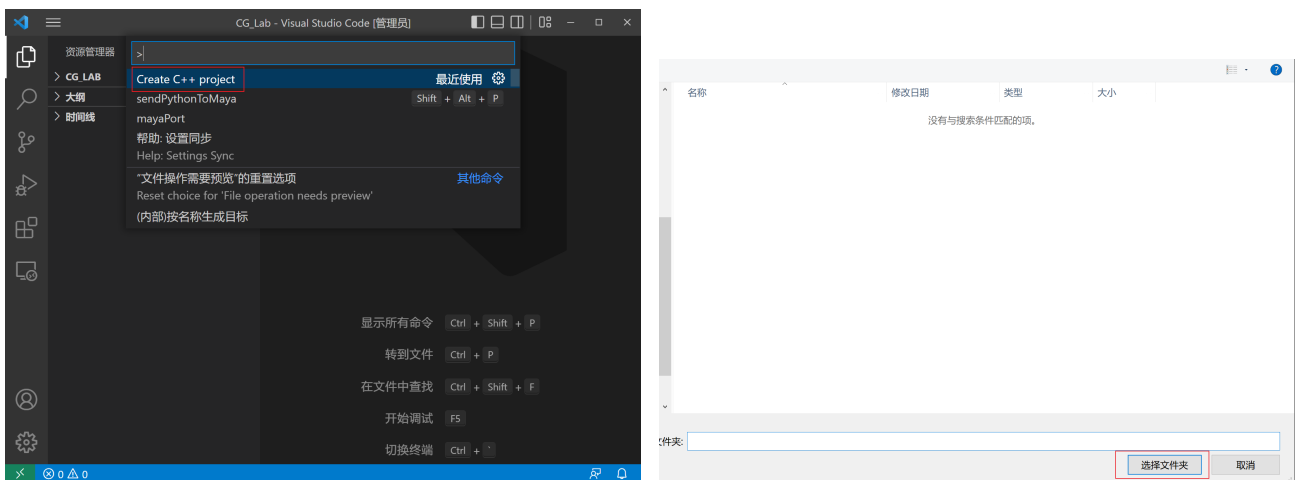


(a) 标记

(b) 安装

Figure 3.3: MinGW Installation Manager

3. 点击 Ctrl+Shift+P，搜索 Create C++ project，点击后在弹出窗口点击选择文件夹，生成项目文件，见图 3.4。
4. 在终端输入 make run，成功输出"Hello World" 则配置成功。



(a) 搜索

(b) 生成项目文件

Figure 3.4: VSCode

3.3 glfw 下载

1. 在 CMD 中输入 mingw32-make -v 查看 mingw 版本，如果输出信息有 i686 说明是 32 位版本，如果输出信息包含 x86_64 说明是 64 位版本。
2. 前往 [glfw 官网](#) 下载 Windows pre-compiled binaries，根据 MinGW 选择 64 位或 32 位版本，见图 3.5。
3. 解压 glfw 压缩包。
4. 将 include\GLFW 文件夹复制到 CG_Lab\include\下。
5. 将 lib-mingw\下的 libglfw3.a 和 libglfw3dll.a 复制至 CG_Lab\lib\下。

6. 将 lib-mingw\下的 glfw3.dll 复制到 CG_Lab\output\文件夹下。

Windows pre-compiled binaries

These packages contain the GLFW header files, [documentation](#) and release mode static libraries, DLLs and import libraries for Visual C++ 2010-2019 and the 2022 preview, MinGW-w64 and plain MinGW.

Binaries for Visual C++ 2010 and plain MinGW are only available in the 32-bit package.

64-bit Windows binaries

32-bit Windows binaries

Figure 3.5: glfw

3.4 glad 库

1. 前往[glad 在线服务](#)生成静态库，见图 3.6，生成后下载 glad.zip 文件。

The screenshot shows the glad online service interface. It includes sections for Language (C/C++), Specification (OpenGL), API (gl, Version 4.6), Profile (Core), and various extensions (gles1, gles2, glsc2). There is a search bar for extensions and a list of available extensions. At the bottom, there are options to generate a loader, omit KHR, and use local files. A GENERATE button is located at the bottom right.

Figure 3.6: glad

2. 解压文件夹。
3. 右键在终端打开，输入命令 `gcc ./src/glad.c -c -I ./include/`
4. 成功后继续输入命令 `ar -rc libglad.a glad.o`
5. 将生成的 libglad.a 文件复制到 CG_Lab\lib\下。
6. 将 include\glad\和 include\KHR\文件夹复制到 CG_Lab\include\下。
7. 最后文件夹结构如图 3.7。

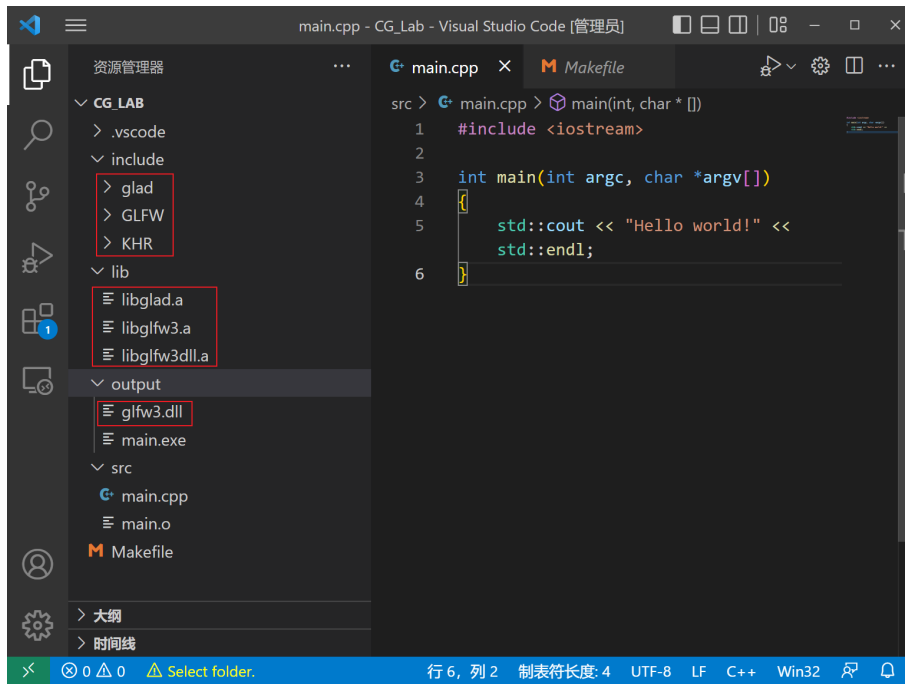


Figure 3.7: 文件图

3.5 修改 Makefile 文件

1. 下载[Makefile 文件](#)替换旧 Makefile 文件。

3.6 测试

1. 下载[HelloWindow.cpp 文件](#)至 src\hello_window。
2. 终端运行 `make run dir=hello_window`，运行成功即测试通过。(make 报错则使用 `mingw32-make run dir=hello_window`)
3. 测试结果见图 3.8。
4. 注：可以使用 `make clean dir=xxx` 命令来清除编译结果。

3.7 更多教程。

- [Visual Studio 配置 OpenGL 环境。](#)
- [Visual Studio 2019 配置 OpenGL](#)

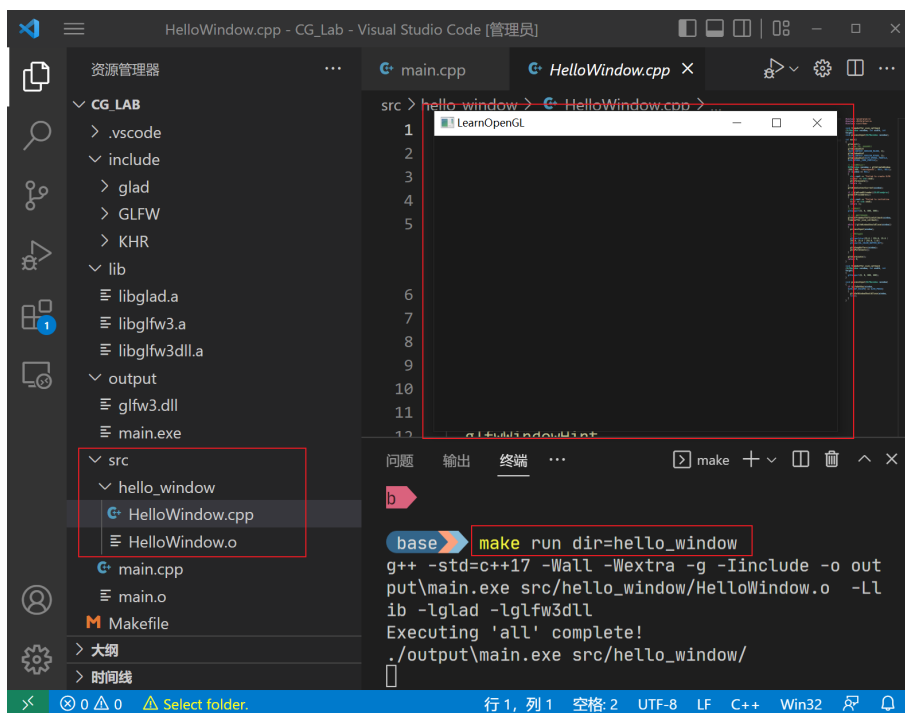


Figure 3.8: 测试结果