华东师范大学计算机科学与技术实验报告

实验课程：计算机图形学	年级：2018	实验成绩：
实验名称：熟悉 OpenGL 与 GLFW 的基本框架以及 shader 基本操作	姓名：李泽浩	实验日期：2021/3/23
实验编号：3	学号：10185102142	实验时间：13:00-14:40
指导教师：李洋	组号：	

一、实验目的

利用操作系统 API 实现基本三角形绘制功能

二、实验环境

分别在windows+visual studio下C++实现 和 macOS下利用python实现

由于Python实现太过简单，直接把代码贴在最后，中间实现步骤都为C++实现过程

三、实验内容

1.编辑 build.bat 文件，生成与本机 Visual Studio 版本相匹配的工程;

2. 进入上一个步骤生成的目录下，点击 *.sln 文件进入项目;
3. 将 Hello Triangle 设置为启动项目;
4. 输出显示 OpenGL 的版本号;
5. 绘制三角形。

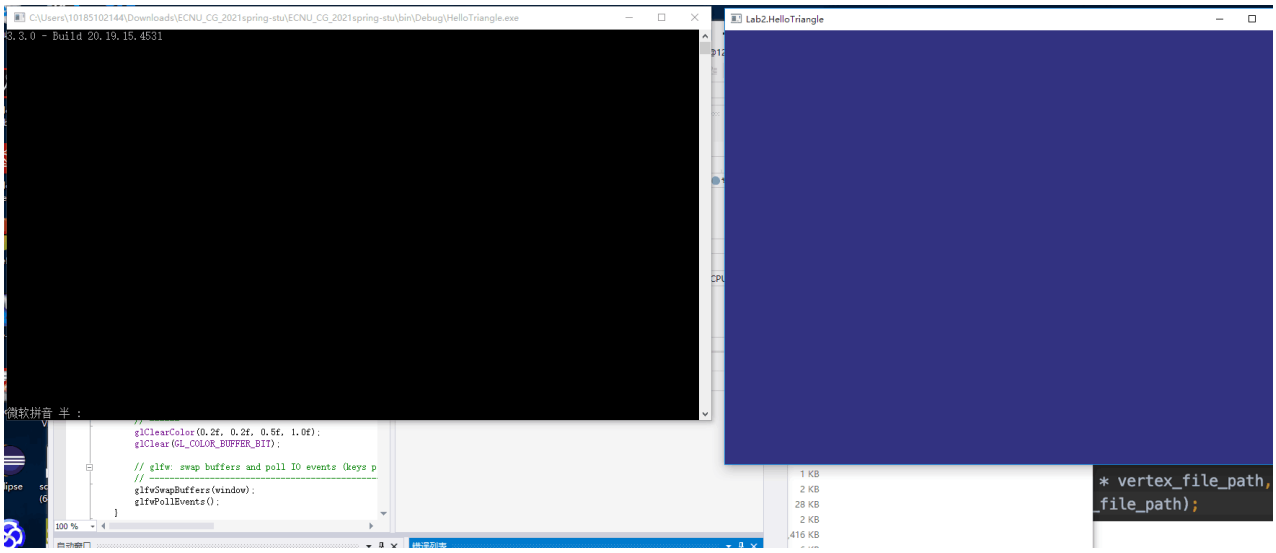
四、实验过程与分析

1.通过Cmake创建项目并配置OpenGL环境

```
mkdir vs2010
cd vs2010
.\..\cmake\bin\cmake.exe .. -G "Visual Studio 16 2019"
cd ..
```

2.获得版本号并输出

```
const GLubyte* version = glGetString(GL_VERSION);
printf("%s\n", version);
```



3.三角形的绘制

根据PP T步骤进行代码编写

前几步没有完全理解，照搬了PPT上代码

```
const char * vertex_file_path = "simple.vs";
const char * fragment_file_path = "simple.fs";
GLuint g = Loshader(vertex_file_path,fragment_file_path);
Gluint VBO = 0;
glGenBuffers(1,&VBO);
glBindBuffer(GL_ARRAY_BUFFER,VBO);
GlbufferData(GL_ARRAY_BUFFER,sizeof(vertices),vertices,GL_STATIC_DRAW);
```

```
//vertex_shader
const char* vertex_shader =
    "#version 400\n"
    "in vec3 vp;"
    "void main() {"
    " gl_Position = vec4(vp, 1.0);" "}";

//fragment_sahder
const char* fragment_shader =
    "#version 400\n"
    "out vec4 frag_colour;"
    "void main(){"
    " frag_colour = vec4(0.5, 0.0, 0.5, 1.0);" "}";
```

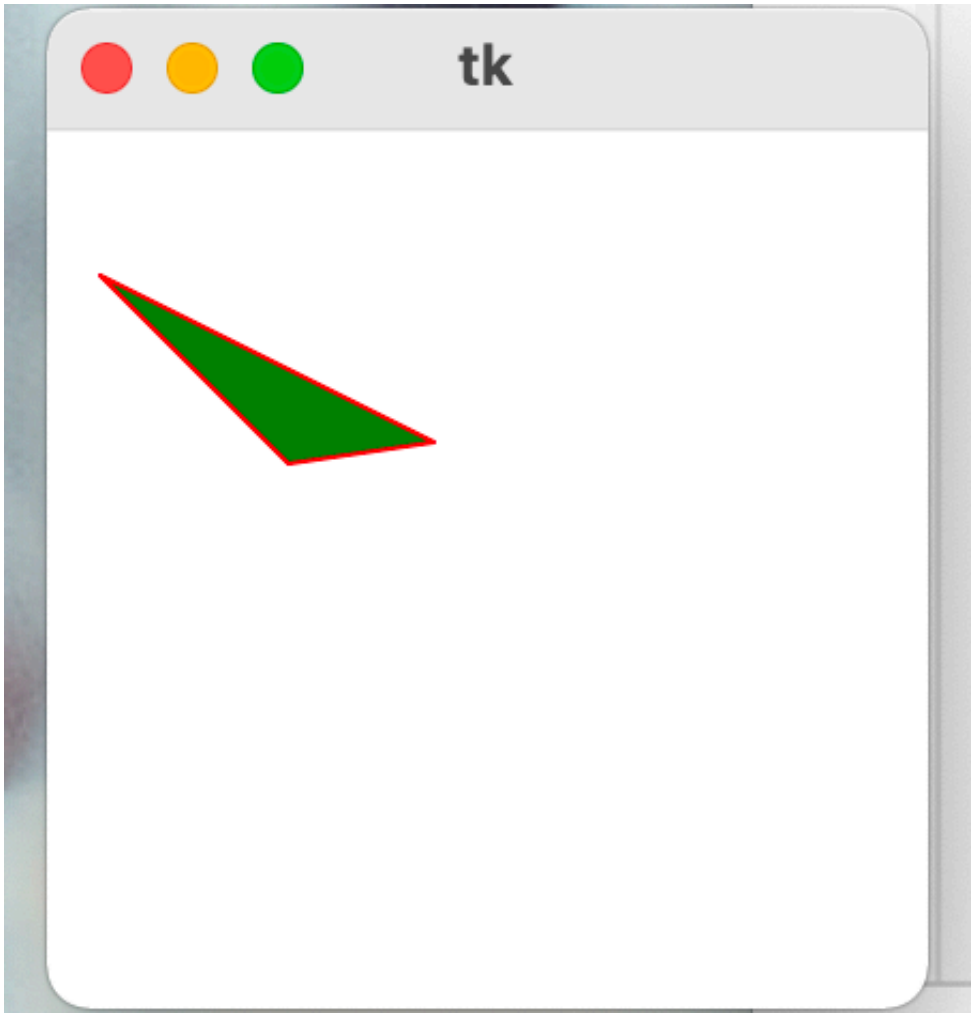
```
//编译+链接
GLuint vs = glCreateShader(GL_VERTEX_SHADER);
glShaderSource(vs, 1, &vertex_shader, NULL);
glCompileShader(vs);
GLuint fs = glCreateShader(GL_FRAGMENT_SHADER);
glShaderSource(fs, 1, &fragment_shader, NULL);
glCompileShader(fs);

GLuint shader_programme = glCreateProgram();
glAttachShader(shader_programme, fs);
glAttachShader(shader_programme, vs);
glLinkProgram(shader_programme);

//创建Vertex Array
GLuint vao = 0;
glGenVertexArrays(1, &vao);
glBindVertexArray(vao);
glEnableVertexAttribArray(0);

//创建Vertex Buffer
GLuint vbo = 0;
glGenBuffers(1, &vbo);
glBindBuffer(GL_ARRAY_BUFFER, vbo);
glBufferData(GL_ARRAY_BUFFER, 9 * sizeof(float), points, GL_STATIC_DRAW);
glBindBuffer(GL_ARRAY_BUFFER, vbo);
glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 0, NULL);
```

五、实验过程总结

Python实现代码如下：

```
from tkinter import *

def main():
    root = Tk()

    w = Canvas(
        root,
        width=200,
        height=200,
        background="white"
    )
    w.pack()

    #输入三角形三点的坐标
    x0,y0,x1,y1,x2,y2 = map(int,input().split(" "))

    points = [x0,y0,x1,y1,x2,y2]

    # 根据点来连线
    w.create_polygon(
        points,
        outline="red",  # 线的颜色
        fill="green"  # 填充色
    )
    mainloop()

if __name__ == "__main__":
    main()
```

```
12 34 56 78 90 73
```



六、附录