一篇就够的超良心pyOpenGL入门教程，不香喷我！

OpenGL （python版本）

OpenGL是用于渲染2D、3D 矢量图 形的扩语言、跨平台的应用程序编程接口。这个接口由近350个不同的函数调用组成，用来从简单的图形比特绘制复杂的三维景象。而另一种程序接口系统是仅用于Microsoft Windows上的Direct3D。OpenGL常用于CAD、虚拟实境、科学可视化程序和电子游戏开发。博客内容是基于python OpenGL进行讲解，感觉C++有点难，等到后面有时间了再学习，不过都是想通的，我是基于C++教程或网站整理的，对于python开发者或者C++开发者都是适用的。

划重点了！！！

这次真的很用心在做这个教程，绘图整理不易希望**大家点赞，关注一波！！！**，这样我才能更加努力创作呀！！！

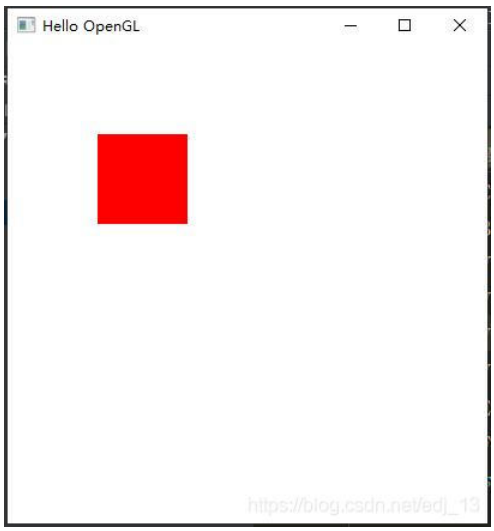
1.第一个例子

首先在上一小节安装好OpenGL后我们先不管其他的先运行一个例子看看效果，对于这个例子的细节我们可以先忽略，我就简单把它作为后面讲解的模板了，废话不多说先上代码：

```
import numpy as np
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
class Demo(object):
    def __init__(self):
        # self.geometry = geometry
        glutInit()  # 启动glut
        glutInitDisplayMode(GLUT_SINGLE | GLUT_RGBA)
        glutInitWindowSize(400, 400)
        glutCreateWindow(b"Hello OpenGL")  # 设定窗口标题
        glutDisplayFunc(self.draw_geometry)  # 调用函数绘制
        self.init_condition()  # 设定背景
        glutMainLoop()

    def init_condition(self):
        glClearColor(1.0, 1.0, 1.0, 1.0)  # 定义背景为白色
        gluOrtho2D(-8.0, 8.0, -8.0, 8.0)  # 定义xy轴范围
    def render(self):
        pass
    def draw_geometry(self):
        glClear(GL_COLOR_BUFFER_BIT)
        glColor3f(1.0, 0.0, 0.0)  # 设定颜色RGB
        glBegin(GL_QUADS)
        glVertex2f(-2, 2)
        glVertex2f(-2, 5)
        glVertex2f(-5, 5)
        glVertex2f(-5, 2)
        glEnd()
        glFlush()  # 执行绘图
if __name__ == "__main__":
    Demo()
```

运行上述代码后相应的显示为：



通过上面的例子我们简单运行了一个例子，请大家将代码拷贝为你的模板，将在后面学习中继续使用呢，好了接下来就开始我们的OpenGL学习之旅吧！

2.基本属性

绘制一个基本的二维或三维图形都是从最基本的坐标系统和绘制点、线和多边形开始，首先需要了解这些的基本约定，这一节讨论xy平面或xyz空间固定位置的绘制图形。

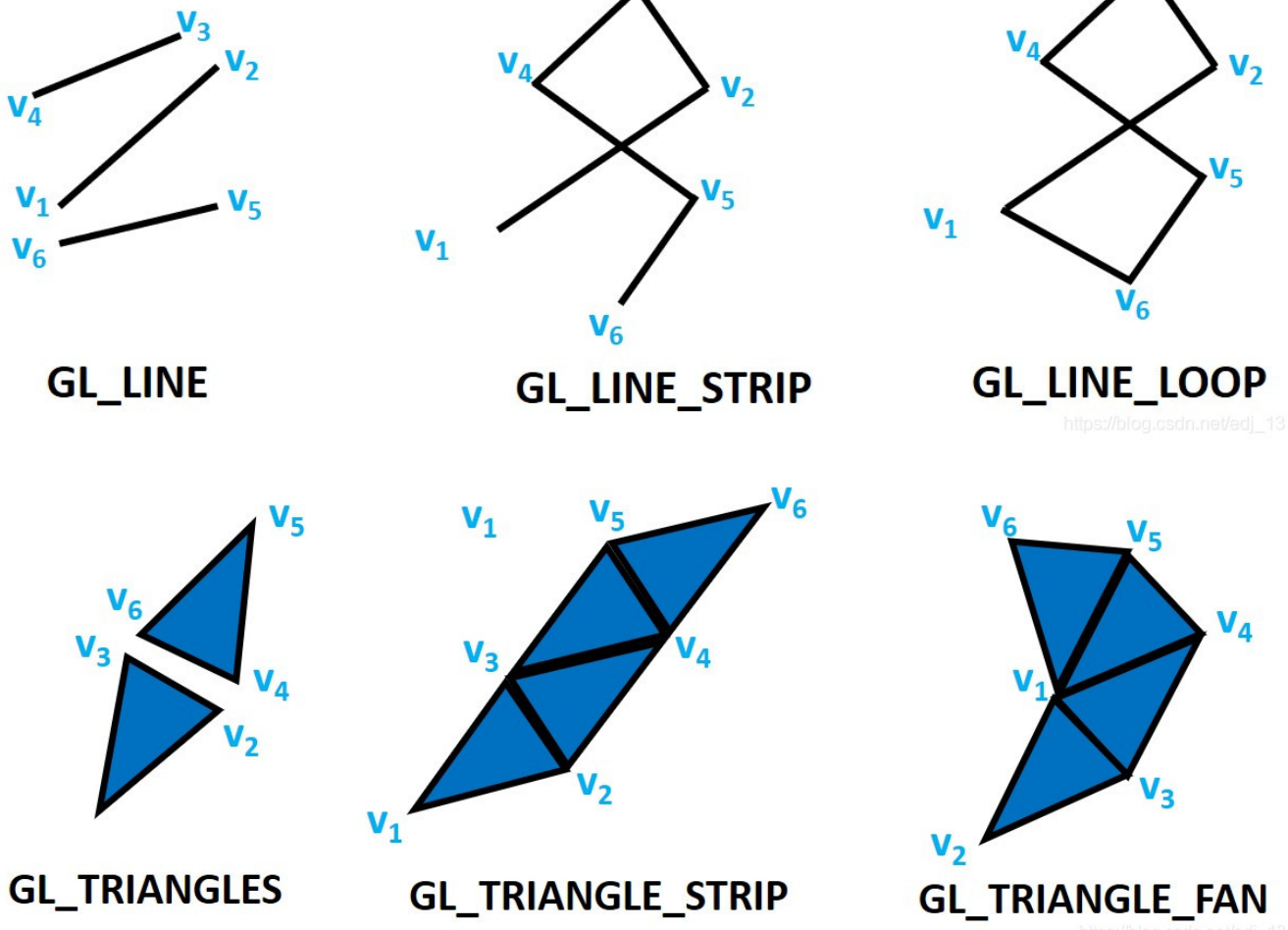
注意：**所有调用glVertex*的命令必须通过OpenGL的命令glBegin()和glEnd()括起来**，这两个命令给出了绘图的信号。

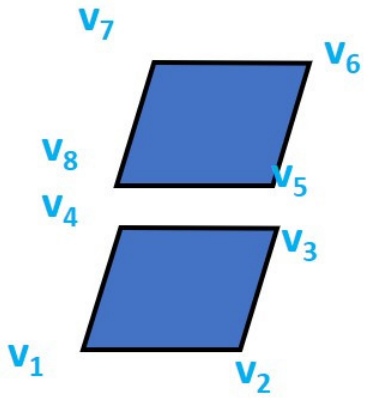
相应的语法为：

```
glBegin(mode)
.....
glEnd()
```

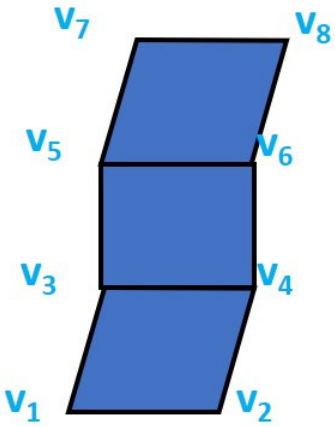
相应的mode为由矢量指定的类型，总共可以接受10个符号常数，分别为：
GL_POINTS,**GL_LINES**,**GL_LINE_STRIP**,**GL_LINE_LOOP**,**GL_TRIANGLES**,**GL_TRIANGLE_STRIP**,**GL_TRIANGLE_FAN**,**GL_QUADS**,**GL_QUAD_STRIP**和**GL_POLYGON**。

上述不同的mode参数的意义可通过下面的示意图给出相应的解释：

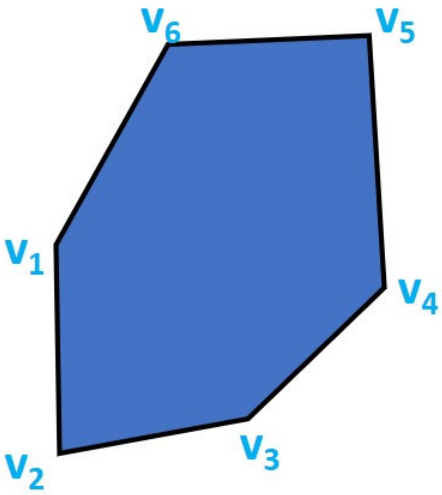




GL_QUADS



GL_QUAD_STRIP



GL_POLYGON

通过上面的几个示意图就可以清晰了解我们在绘图时需要选择的mode是啥了，是不是感觉超级简单，感觉这一块内容下面的都不要看了呢！

- 点

通过下面的语句可画出三个点：

```
glBegin(GL_POINTS)
glVertex2f(1.0, 1.0)
glVertex2f(2.0, 1.0)
glVertex2f(2.0, 2.0)
glEnd()
```

将其写入一个函数利用 **glutDisplayFunc(self.draw_point)**调用，相应函数如下：

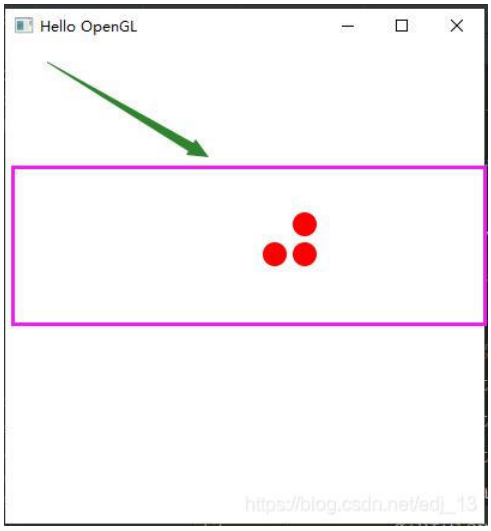
```
def draw_point(self):
    glClear(GL_COLOR_BUFFER_BIT)
    glColor3f(1.0, 0.0, 0.0)  # 设定颜色RGB
    glBegin(GL_POINTS)
    glVertex2f(1.0, 1.0)
    glVertex2f(2.0, 1.0)
    glVertex2f(2.0, 2.0)
    glEnd()
    glFlush()
```

通过运行我们发现绘制的点非常小，几乎看不见，我们可通过调用下面函数来显示点的大小：

```
glPointSize(n)
#Points are n pixels in diameter
glEnable(GL_POINT_SMOOTH)
```

```
glHint(GL_POINT_SMOOTH_HINT,GL_NICEST)
glEnable(GL_BLEND)
glBlendFunc(GL_SRC_ALPHA,GL_ONE_MINUS_SRC_ALPHA)
```

最终效果图为：



- 线

类似的线的绘制命令如下：

```
glBegin(GL_LINES)
glVertex3f(x1,y1,z1)
glVertex3f(x2,y2,z2)
glVertex3f(x3,y3,z3)
glVertex3f(x4,y4,z4)
glEnd()
```

当然如果你嫌画的线条太细，可通过下面下面命令平滑和加宽线条：

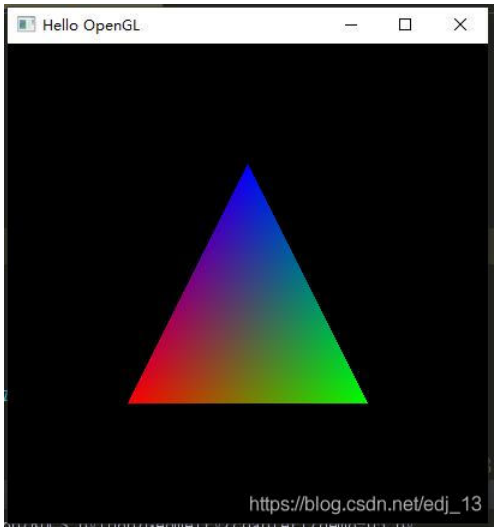
```
glLineWidth(n)    #Lines are n pixels wide
glEnable(GL_Line_SMOOTH)
glHint(GL_Line_SMOOTH_HINT,GL_NICEST)
glEnable(GL_BLEND)
glBlendFunc(GL_SRC_ALPHA,GL_ONE_MINUS_SRC_ALPHA)
```

- 多边形

类似的通过下面的命令绘制三角形：

```
glBegin(GL_TRIANGLES)
glVertex3f(x1,y1,z1)
glVertex3f(x2,y2,z2)
glVertex3f(x3,y3,z3)
glEnd()
```

下面通过绘制一个三角形来加深我们的理解，具体效果如下：



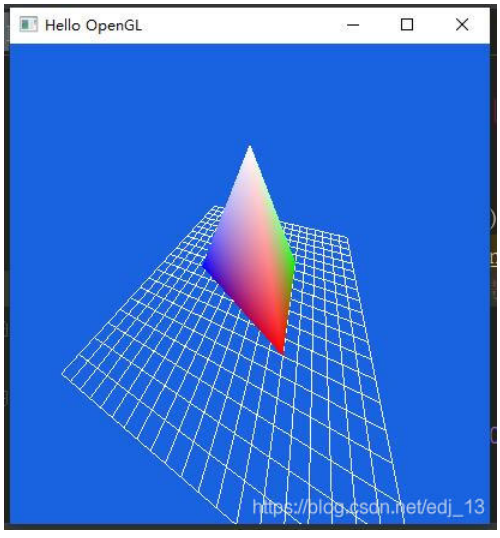
相应的定义三角形的函数式为：

```
def draw_triangles(self):
    glClear(GL_COLOR_BUFFER_BIT)
    glBegin(GL_TRIANGLES)
    glColor3f(1,0,0)
    glVertex3f(-0.5,-0.5,0.0)
    glColor3f(0, 1, 0)
    glVertex3f(0.5, -0.5, 0)
    glColor3f(0, 0, 1)
    glVertex3f(0, 0.5, 0)
    glEnd()
    glFlush()
```

而绘制四边形命令为：

```
glBegin(GL_QUADS)
glVertex3f(x1,y1,z1)
......
glVertex3f(xn,yn,zn)
glEnd()
```

现在我们再绘制一个实例来加深我们的理解，相应效果图为：



具体的绘制图形的函数式为：

```
def draw_polygon(self):
    glClear(GL_COLOR_BUFFER_BIT)
    glColor3f(1.0, 1.0, 1.0)  # 设定颜色RGB
    glBegin(GL_LINES)
    datas = [-2.5+i*0.25 for i in range(21)]
    for i in datas:
    glVertex3f(i,0,2.5)
    glVertex3f(i,0,-2.5)
    glVertex3f(2.5,0,i)
```

```
    glVertex3f(-2.5,0,i)
    glEnd()

    glBegin(GL_TRIANGLE_STRIP)
    glColor3f(1,1,1)
    glVertex3f(0,2,0)
    glColor3f(1, 0, 0)
    glVertex3f(-1, 0, 1)
    glColor3f(0, 1, 0)
    glVertex3f(1, 0, 1)
    glColor3f(0, 0, 1)
    glVertex3f(0, 0, -1.4)
    glColor3f(1, 1, 1)
    glVertex3f(0, 2, 0)
    glColor3f(1, 0, 0)
    glVertex3f(-1, 0, 1)
    glEnd()
    glFlush()
```

通过这一小节的学习我们熟悉了PyOpenGL中基本元素的绘制，可以利用基本模板绘制图形，而OpenGL的学习之旅才真正的开始，涉及到的东西还比较多，下面大家可以自己尝试构建一下下面的实例，具体实现后面揭晓。

课后作业：