컴퓨터그래픽스 실습1

2016133 이유진

import math, numpy  
import OpenGL  
  
  
SCR\_WIDTH = 800  
SCR\_HEIGHT = 600  
  
  
vertexData = numpy.array(  
 [-0.200, 0.234, -0.089,  
 -0.771, 0.254, -0.089,  
 -0.000, 0.003, -0.089,  
 -0.771, 0.254, -0.089,  
 -0.324, -0.147, -0.089,  
 -0.000, 0.003, -0.089,  
 -0.324, -0.147, -0.089,  
 -0.477, -0.653, -0.089,  
 -0.000, 0.003, -0.089,  
 -0.477, -0.653, -0.089,  
 -0.000, -0.383, -0.089,  
 -0.000, 0.003, -0.089,  
 -0.000, -0.383, -0.089,  
 0.477, -0.654, -0.089,  
 -0.000, 0.003, -0.089,  
 0.477, -0.654, -0.089,  
 0.324, -0.148, -0.089,  
 -0.000, 0.003, -0.089,  
 0.324, -0.148, -0.089,  
 0.772, 0.253, -0.089,  
 -0.000, 0.003, -0.089,  
 0.772, 0.253, -0.089,  
 0.200, 0.234, -0.089,  
 -0.000, 0.003, -0.089,  
 0.200, 0.234, -0.089,  
 0.000, 0.814, -0.089,  
 -0.000, 0.003, -0.089,  
 0.000, 0.814, -0.089,  
 -0.200, 0.234, -0.089,  
 -0.000, 0.003, -0.089  
 ], numpy.float32)  
  
vertexShaderSource = """  
#version 330 core  
layout (location = 0) in vec3 aPos;  
  
void main(){  
 gl\_Position = vec4(aPos.x, aPos.y, aPos.z, 1.0);  
}  
"""  
fragmentShaderSource = """  
#version 330 core  
out vec4 FragColor;  
  
void main(){  
 FragColor = vec4(0.0f, 0.0f, 1.0f, 1.0f);  
}  
"""  
fragmentShaderSource2 = """  
#version 330 core  
out vec4 FragColor;  
  
void main(){  
 FragColor = vec4(1.0f, 1.0f, 1.0f, 1.0f);  
}  
"""  
  
from glfw import (window\_hint, init, create\_window, terminate,  
 make\_context\_current, swap\_buffers, poll\_events,  
 window\_should\_close,  
 CONTEXT\_VERSION\_MAJOR,  
 CONTEXT\_VERSION\_MINOR, OPENGL\_FORWARD\_COMPAT,  
 OPENGL\_PROFILE, OPENGL\_CORE\_PROFILE)  
  
  
from OpenGL.GL import ( GL\_TRUE, GL\_COLOR, GL\_POINTS, GL\_LINE\_LOOP,GL\_ARRAY\_BUFFER,GL\_STATIC\_DRAW,  
 GL\_LIGHTING,GL\_DEPTH\_TEST,GL\_TRIANGLES,GL\_FRONT\_AND\_BACK,GL\_LINE,  
 glPolygonMode,glPointSize,  
 glClearBufferfv, glGenBuffers,glBindBuffer,glBufferData,glVertexAttribPointer,  
 glDrawArrays, glLinkProgram,glEnableVertexAttribArray,GL\_FLOAT, GL\_FALSE,glDeleteProgram,  
 glGenVertexArrays, glBindVertexArray,glDeleteBuffers , glDeleteVertexArrays,  
 glLineWidth,glDisable, glColor3f)  
  
from OpenGL.GL.shaders import (GL\_VERTEX\_SHADER, GL\_FRAGMENT\_SHADER,  
 compileShader, glCreateProgram,  
 glAttachShader, glUseProgram, glGetUniformLocation, glUniform1f)  
  
  
def main():  
 # Initialize the library  
 if not init():  
 return  
  
 window\_hint(CONTEXT\_VERSION\_MAJOR, 3)  
 window\_hint(CONTEXT\_VERSION\_MINOR, 3)  
 window\_hint(OPENGL\_FORWARD\_COMPAT, GL\_TRUE)  
 window\_hint(OPENGL\_PROFILE, OPENGL\_CORE\_PROFILE)  
  
 # program = glutils.loadShaders(vs\_source, fs\_source)  
 # Create a windowed mode window and its OpenGL context  
 window = create\_window(SCR\_WIDTH, SCR\_HEIGHT, "Window Only", None, None)  
 if not window:  
 terminate()  
 return  
  
 # Make the window's context current  
 make\_context\_current(window)  
  
 shaderV = compileShader([vertexShaderSource], GL\_VERTEX\_SHADER)  
 shaderF = compileShader([fragmentShaderSource], GL\_FRAGMENT\_SHADER)  
 program = glCreateProgram()  
  
  
 glAttachShader(program, shaderV)  
 glAttachShader(program, shaderF)  
 glLinkProgram(program)  
  
 # ----------- NEW LINE  
 shaderF2 = compileShader([fragmentShaderSource2], GL\_FRAGMENT\_SHADER)  
 program2 = glCreateProgram()  
  
 glAttachShader(program2, shaderV)  
 glAttachShader(program2, shaderF2)  
 glLinkProgram(program2)  
 # ----------- NEW END  
  
 vao = glGenVertexArrays(1)  
 glBindVertexArray(vao)  
 vertexBuffer = glGenBuffers(1)  
 glBindBuffer(GL\_ARRAY\_BUFFER, vertexBuffer)  
  
 glBufferData(GL\_ARRAY\_BUFFER, 4 \* len(vertexData), vertexData,  
 GL\_STATIC\_DRAW)  
 # enable vertex array  
 glEnableVertexAttribArray(0)  
 # set buffer data pointer  
 glVertexAttribPointer(0, 3, GL\_FLOAT, GL\_FALSE, 0, None)  
 # unbind VAO  
 glBindVertexArray(0)  
  
  
 # Loop until the user closes the window  
 while not window\_should\_close(window):  
 # Render here, e.g. using pyOpenGL  
  
 glClearBufferfv(GL\_COLOR, 0, (0.0, 0.0, 0.0, 1.0))  
  
 glUseProgram(program)  
 glBindVertexArray(vao)  
 glDisable(GL\_DEPTH\_TEST)  
  
 #  
 glPointSize(20.0)  
 glDrawArrays(GL\_TRIANGLES, 0, 30)  
  
 glUseProgram(program2)  
 glDrawArrays(GL\_LINE\_LOOP, 0, 30)  
 # Swap front and back buffers  
 swap\_buffers(window)  
  
 # Poll for and process events  
 poll\_events()  
  
 glBindVertexArray(0)  
 glDeleteBuffers(1, [vertexBuffer])  
 glDeleteProgram(program)  
 glDeleteVertexArrays(1, [vao])  
 terminate()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

텍스트, 모니터, 컴퓨터, 노트북이(가) 표시된 사진

자동 생성된 설명

별이 찌그러져서 윈도우 크기를 800,800으로 변경한 실행 결과도 첨부합니다.

텍스트, 컴퓨터, 모니터, 노트북이(가) 표시된 사진

자동 생성된 설명