

# 3D 그래픽스, 셰이더, OpenGL

3D Graphics Programming with OpenGL Shaders

[biztripcru@gmail.com](mailto:biztripcru@gmail.com)

© 2022–2024. [biztripcru@gmail.com](mailto:biztripcru@gmail.com). All rights reserved.  
모든 저작권은 [biztripcru@gmail.com](mailto:biztripcru@gmail.com) 에게 있습니다.

# OpenGL 라이브러리 소개

Introducing the OpenGL library

# Contents 내용

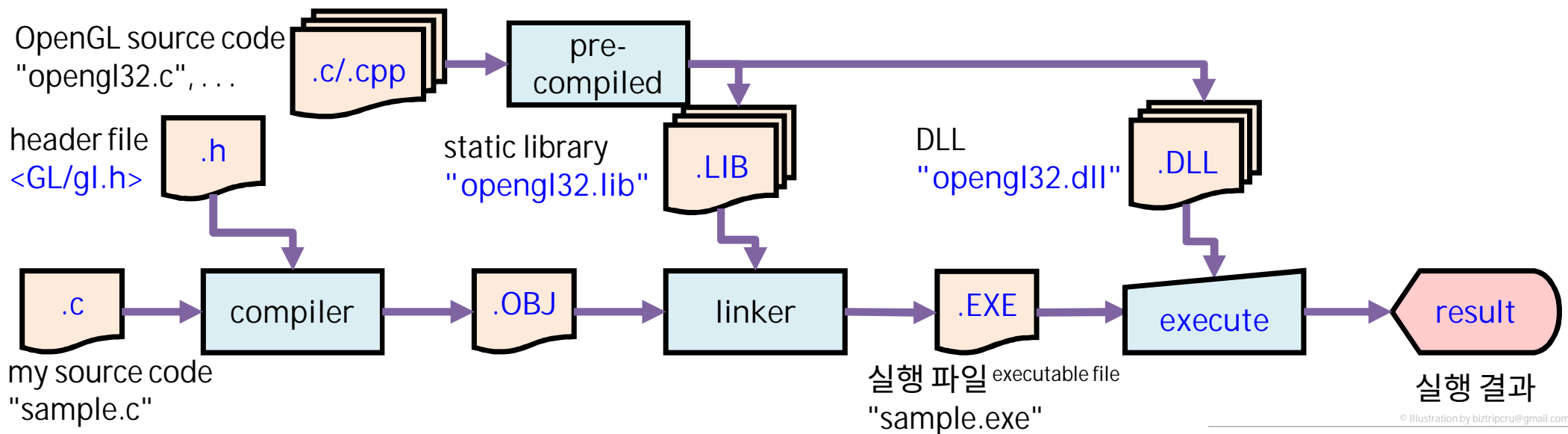
- OpenGL 라이브러리
- 윈도우 시스템
- OpenGL Extension
- OpenGL 라이브러리 특성

# OpenGL 라이브러리

OpenGL 오픈지엘 library

# C/C++ 라이브러리 library

- static library 정적 라이브러리 : 컴파일 시에 결합하는 방식
  - ".lib" : 현재는 기본적인 정보만 저장
- dynamic library 동적 라이브러리 : 실행 시에 결합하는 방식
  - ".so" (shared object, Unix/Linux) 또는 ".dll" (dynamic-link library, Window)



© Illustration by biztripcu@gmail.com

# OpenGL 라이브러리

- OpenGL core library 코어 라이브러리
  - `libGL.so` : 대부분의 Unix/Linux 시스템, Mac 시스템
  - `opengl32.dll` : 마이크로소프트 윈도우 Windows
    - ▶ 윈도우 최초 설치 시는 매우 오래된 버전을 제공
    - ▶ 윈도우 그래픽스 드라이버를 업데이트 하면 최신 버전으로 업데이트됨
- OpenGL Utility Library (GLU) 유틸리티 라이브러리 → deprecated 사용중지
  - `libGLU.so` for Linux or `glu32.dll` for Windows
  - 일부 core library 보강 기능, 예전 코드 legacy code 에서 사용
  - 현재는 사용하지 않음

# 윈도우 시스템

Window System

# 윈도우 시스템의 역할

- 현재 사용되는 윈도우 시스템
  - X window systems on Linux
  - MS windows systems on MS Windows
  - iOS windows systems on Mac
- 윈도우 시스템의 역할
  - 2D 화면 전체의 제어
  - 화면에 2D 윈도우 생성/이동/크기조정/축소/삭제
- OpenGL 입장
  - OpenGL은 2D 윈도우 내에 2D/3D 그래픽 출력
  - 윈도우 자체는 윈도우 시스템이 제어

GNU free document license, 2022.01.08  
<https://commons.wikimedia.org/wiki/File:X-Window-System.gif>



public domain, 2022.01.07  
<https://publicdomainvectors.org/en/free-clipart/Hero-from-video-games/84515.html>



# OpenGL 프로그램과 윈도우 시스템의 연결

- OpenGL 프로그램에서 윈도우 라이브러리 필요
  - 2D 윈도우 생성 필요 → 윈도우와의 연결 라이브러리 사용
  - GLX for X window systems
  - WGL for MS-Windows
  - AGL for Macintosh
- 단일-플랫폼 single-platform OpenGL 프로그램은
  - 해당 윈도우 연결 라이브러리만 사용 → (OpenGL + GLX + XWin)
- 멀티-플랫폼 multi-platform OpenGL 프로그램이라면?
  - 단일 소스 코드 one source code 로
  - 많은 윈도우 시스템/플랫폼을 지원할 방법은?
    - ▶ 많은 if 문으로 처리?

# GLUT deprecated **사용중지**

- OpenGL Utility Toolkit (GLUT)
  - (거의) 모든 윈도우 시스템의 공통 기능만 제공
    - ▶ 윈도우 생성/제거
    - ▶ 마우스/키보드 입력
    - ▶ 간단한 메뉴 생성
    - ▶ 이벤트 처리
  - 장점: 높은 이식성 portability
  - 단점: 제한된 기능
    - ▶ 예: 슬라이드 바 slide bar 제공하지 않음
  - 현재는 **사용중지 deprecated**, 예전 코드 legacy code 에서 사용

capture by biztripcru@gmail.com, 2022.01.08  
<https://www.opengl.org/resources/libraries/glut/>

## GLUT - The OpenGL Utility Toolkit

We direct you to use FreeGLUT found on SourceForge: <http://freeglut.sourceforge.net/>. The original GLUT has been unsupported for 20 years.

# freeglut deprecated 사용중지

- GLUT 의 관리 부재
  - 저작권 문제로, 업데이트 불가
  - OpenGL 새로운 버전과 작동 불능
- freeglut
  - GLUT의 독립적 구현 independent implementation
  - 저작권 문제 해결, 새로운 기능 추가
  - 그러나, 현재는 **사용중지** deprecated



freeglut 데모 화면

# GLFW

- OpenGL Frame Work
  - official site: <http://www.glfw.org/>
- open source, multi-platform library for OpenGL, OpenGL ES, and Vulkan
  - Vulkan = a new low-level 3D graphics and computing API (since 2016)
  - written in C
  - MS-Windows, macOS, and X Window system 모두 지원
- 모든 윈도우 시스템의 공통 기능 제공
  - 윈도우 생성/제거
  - 마우스/키보드 입력
  - 이벤트 처리

# OpenGL Extension

OpenGL 익스텐션

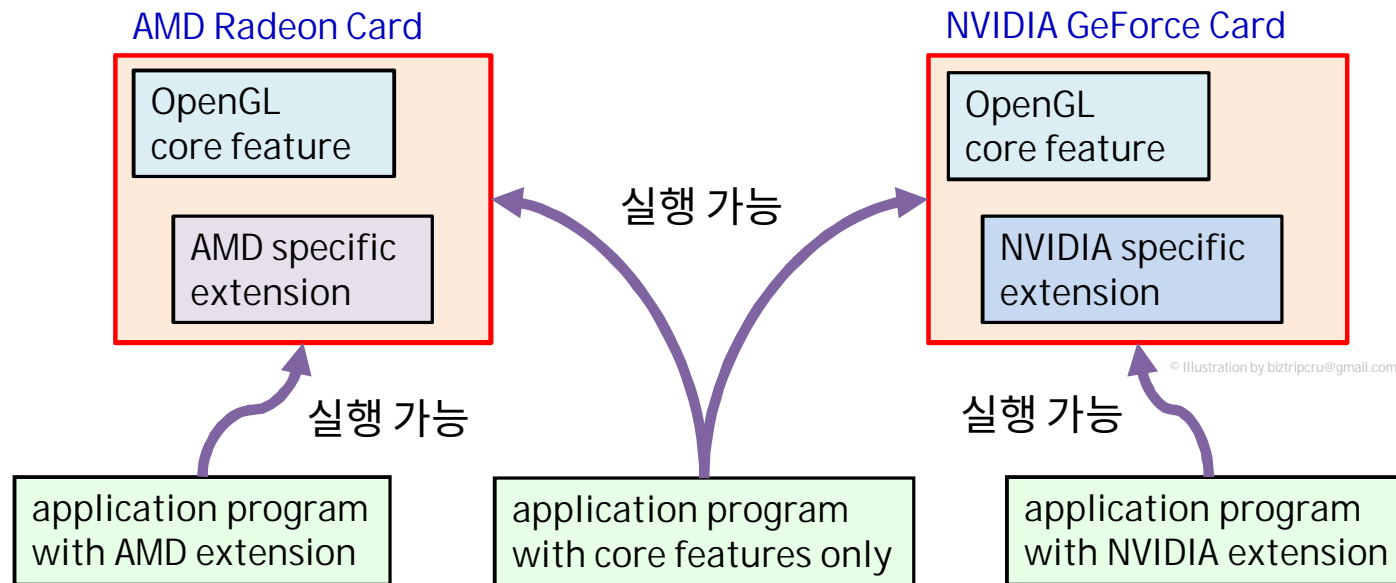
# OpenGL Extensions 익스텐션

- OpenGL core library / core feature
  - OpenGL version X.YY 에서는 **반드시 제공**해야 하는 기능
- OpenGL extensions
  - 핵심 기능 core feature 은 아님 → OpenGL X.YY 가 **제공하지 않아도 문제 없음**
  - 그러나, OpenGL 표준의 일부임 → OpenGL X.YY 가 **제공해도 문제 없음**
  - 현재 400개 이상의 표준 extension 제공
- Why extensions?
  - 기술의 (빠른) 발달 → **하드웨어 제공자가 좋은 기능을 바로 추가**
  - OpenGL core library 는 반드시 필요한 기능이어야 포함시켜 줌
  - 일단 extension 으로 등록 → 나중에 core library 기능으로 포함 가능

# OpenGL Extensions

- OpenGL extension **의 장점**

- core feature만 사용한다면 → **높은 이식성** high portability 확보
- extension 을 사용하면 → 최신 기능을 더 효과적으로 구현 가능



© Illustration by biztripcru@gmail.com

# GLEW 그루

- OpenGL Extension Wrangler Library
  - OpenGL 모든 version 의 사용을 지원
  - OpenGL extension 사용을 도와주는 라이브러리
  - 어떤 extension 이 사용 가능한지 체크 가능
  - 모든 OpenGL version / 표준 extension 의 **모든 함수에 대한 인터페이스 제공**
    - ▶ 해당 함수를 실행 → 하드웨어가 지원하는 경우에만 결과가 나옴
    - ▶ 지원하지 않으면 에러 error 처리
- **사용법**
  - `#include <glew.h>`
  - `glewInit();` // 반드시 실행 !



# GLEW 프로그래밍 예제

- GLEW 에서 특정 OpenGL version / extension 을 선택적으로 사용하는 예제

```
#include <glew.h>
```

```
glwInit(); // 반드시 실행 !
```

```
if (glewIsSupported( "GL_VERSION_4_0" )) {
```

```
    /* use OpenGL version 4.0 features */
```

```
} else {
```

```
    /* use more older OpenGL features */
```

```
}
```

```
if ( glewIsSupported( "GL_ARB_point_sprite" )) {
```

```
    /* use "ARB point sprite" extension features */
```

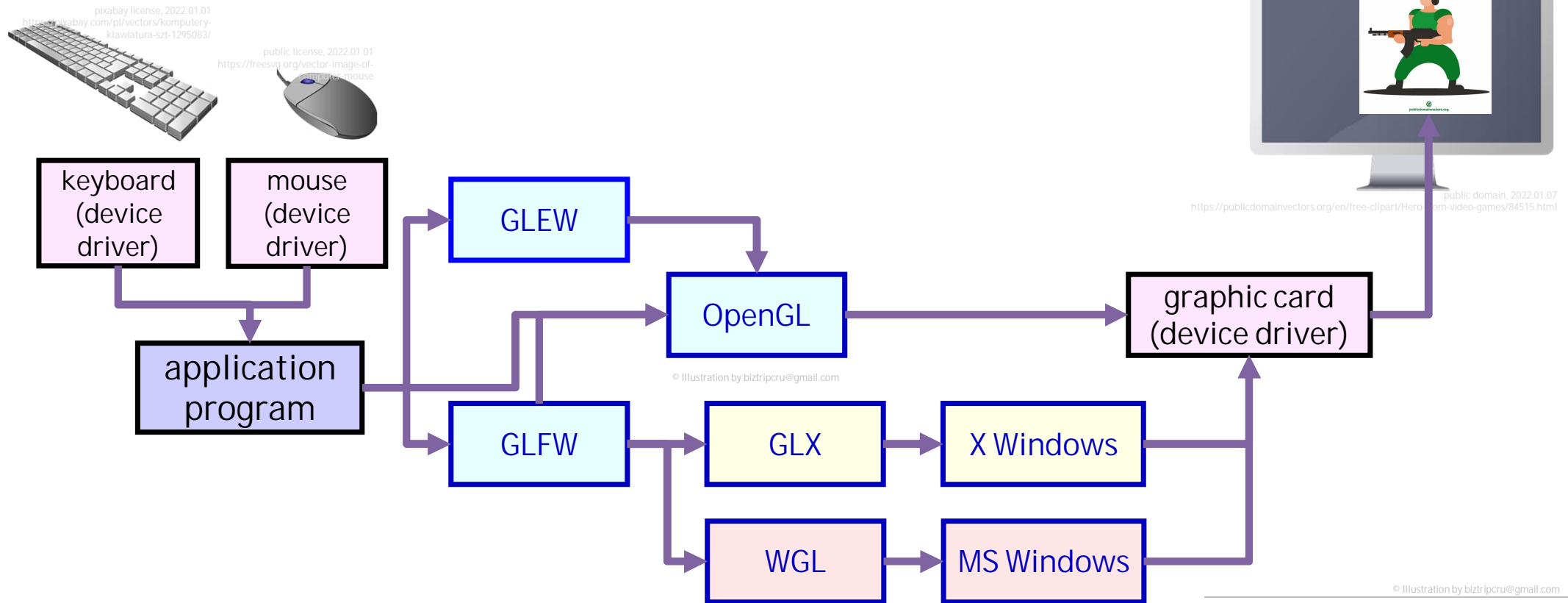
```
} else {
```

```
    /* implement without "ARB point sprite" extension */
```

```
}
```

# OpenGL Software 구성

- 그래픽스 카드 : 매우 복잡한 device driver 디바이스 드라이버 를 가짐

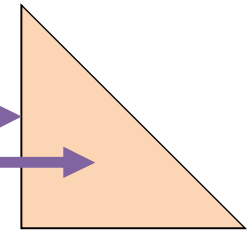


# OpenGL 라이브러리 특성

OpenGL library characteristics

# 3D 그래픽스에서 필요한 기능들

- **프리티비브** primitives **출력**
    - 점 points, 선분 line segments, 삼각형 triangles
  - **속성** attribute **설정**
    - 프리미티브의 속성 : 삼각형의 색상 color
  - **질의** query
    - 현재 상태에 대한 질의
  - **좌표 변환** transformation
    - 모델링 변환, 뷰잉 변환, 프로젝션 변환
- **윈도우 제어** window control : 윈도우 시스템
  - **장치 입력** input : 키보드, 마우스



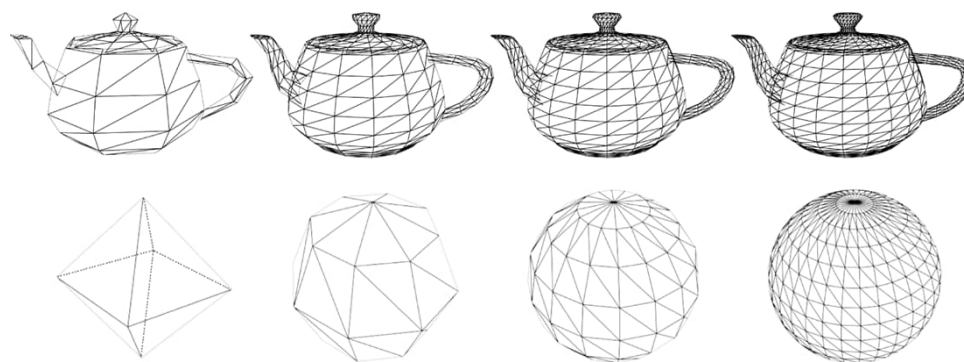
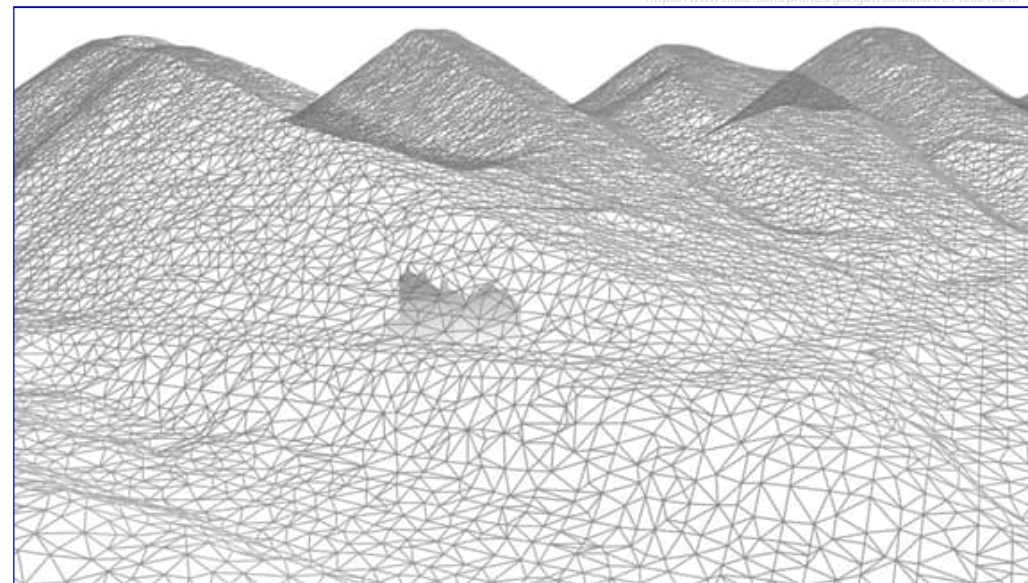
© Illustration by biztripcru@gmail.com

OpenGL 담당

GLFW 담당

# Primitive 출력

- 가장 중요한 primitive = 삼각형
- 삼각형 근사 approximation
  - 많은 물체에 적용 가능
- triangle landscape 랜드스케이프
  - triangle soup
- refinement 정제



simplify 단순화 ←

© 한정현 교수님 (고려대), 2022.01.07  
from "OpenGL ES를 이용한 3차원 컴퓨터 그래픽스 입문" (한글판)

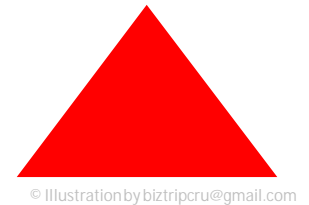
→ refine 정제

# OpenGL state **스테이트** 관리

- OpenGL은 **state machine** 개념으로 구현
  - 내부 상태 state
    - ▶ 예: current color = white
- OpenGL draw **과정**
  - <sup>1</sup> state change : 내부 상태 설정
  - <sup>2</sup> primitive output : 프리미티브 출력
- 같은 primitive 로, 다른 출력 가능 !

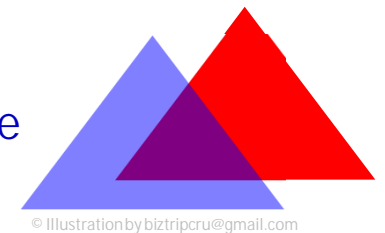
1.1 state: color = red

1.2 draw triangle



2.1 state: color = blue, transparent 투명

2.2 draw triangle



시간

© Illustration by biztripcru@gmail.com

# 객체 지향 object orientation 개념의 결여

- 객체 지향 개념이 전혀 없음

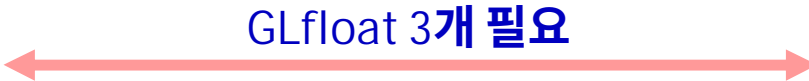

- OpenGL 은 1980년대에 이미 개념 정립

- 불편한 특징: C 언어 기반, function overloading 사용 불가

- C++ 인터페이스도 제안되었으나, 효율성 efficiency 문제로 사용하지 않음
- 사용하려는 자료형에 따라, 함수 이름이 바뀜
  - ▶ `void glUniform3i(GLint location, GLint v0, GLint v1, GLint v2);`
  - ▶ `void glUniform3f(GLint location, GLfloat v0, GLfloat v1, GLfloat v2);`
  - ▶ `void glUniform3iv(GLint location, GLsizei count, const GLint* value);`
  - ▶ `void glUniform3fv(GLint location, GLsizei count, const GLfloat* value);`

# OpenGL function 함수 작명법

- gl / glfw / glew : OpenGL / GLFW / GLEW library
- function name 함수 이름 : 기능 설명
- 2/3/4 : dimension 차원, 필요한 argument 숫자
- i/ui/f/d : data type 자료형 ('f' means 'GLfloat')
- v : vector (또는 array 배열)

- void glUniform3f(GLint location, GLfloat v0, GLfloat v1, GLfloat v2);  

- void glUniform3fv(GLint location, GLsizei count, const GLfloat\* value);  




# OpenGL suffixes 접미어

© Illustration by biztripcru@gmail.com

suffix	data type	OpenGL type	C language
b	8-bit integer	GLbyte	signed char
s	16-bit integer	GLshort	short
i	32-bit integer	GLint, GLsizei	int / long
f	32-bit floating point	GLfloat, GLclampf	float
d	64-bit floating point	GLdouble, GLclampd	double
ub	8-bit unsigned int	GLubyte, GLboolean	unsigned char
us	16-bit unsigned int	GLushort	unsigned short
ui	32-bit unsigned int	GLuint, GLenum, GLbitfield	unsigned int / unsigned long

# OpenGL data types, constants

- OpenGL data types, constants :
  - `#include <GL/glew.h>`
  - `#include <GLFW/glfw3.h>`
  - 실제로는 `<GL/gl.h>`, `<GLES/gles2.h>` 등에 들어 있지만,
  - 위의 `#include` 들이 **자동으로 가져옴**
- 사용 가능한 data type 자료형 : `GLint`, `GLuint`, `GLfloat`, `GLdouble`, ...
- 사용 가능한 constant 상수값: `GL_COLOR_BUFFER_BIT`, `GL_DEPTH_TEST`, ...
  - 항상 `GL_` 로 시작하고, 모두 **영어 대문자 + '\_'** 형태

# Contents **내용**

- OpenGL **라이브러리**
- **윈도우 시스템**
- OpenGL Extension
- OpenGL **라이브러리 특성**

# OpenGL 라이브러리 소개

Introducing the OpenGL library

본고딕 Noto Sans KR

Source Sans Pro

Source Serif Pro

정참판 양반댁 규수 큰 교자 타고 혼례 치른 날

The quick brown fox jumps over the lazy dog

Mathematical Notations  $O(n \log n)$