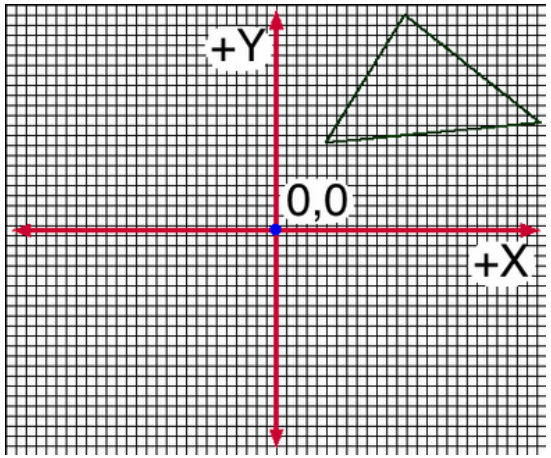
Graphics Programming

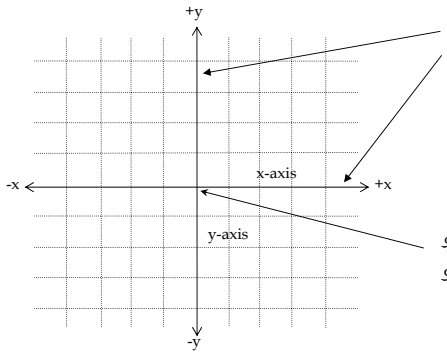
321190
2007년 봄학기
3/13/2007
박경신

Coordinate Systems



2D Cartesian Coordinate Systems

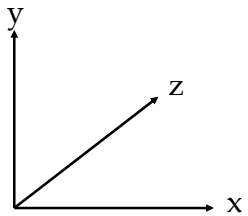
- Cartesian Coordination Systems



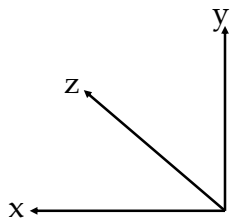
Two axes: **x-axis** and **y-axis**, two straight lines perpendicular to each other, both pass through origin and extends infinitely in two opposite directions

원점 (Origin)은 좌표계의 중심에 위치하고 있고 값은 (0, 0)이다.

3D Cartesian Coordinate Systems

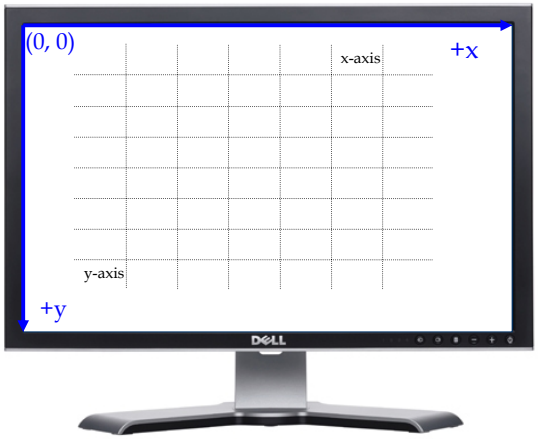


- 왼손 좌표계 (Left-handed coordinate system)는 x+는 오른쪽, y+는 위쪽, z+는 화면안쪽.



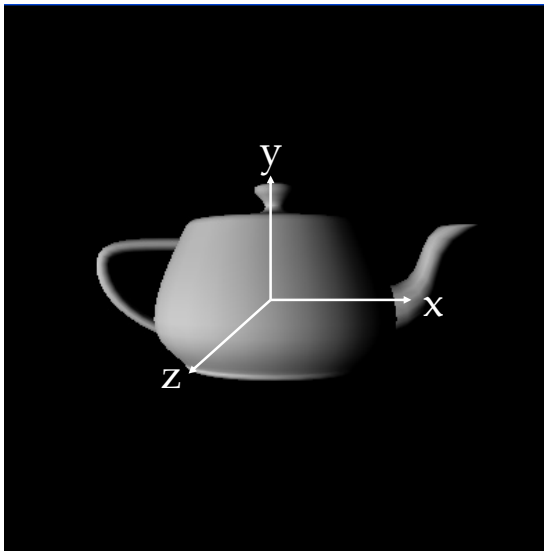
- 오른손 좌표계 (Right-handed coordinate system)는 x+는 왼쪽, y+는 위쪽, z+는 화면안쪽.

Screen Coordinate System



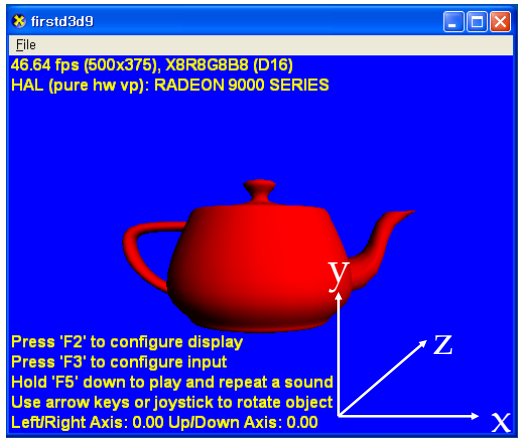
- Screen coordinate system은 원점 (Origin)이 화면의 좌측상단에 위치하고 값은 (0, 0)이다. x+ 오른쪽. y+ 아래쪽.
- 1 unit = 1 pixel

3D Coordinate Systems



- OpenGL은 오른손 좌표계 (Right-handed coordinate system)
- x+ 오른쪽. y+ 위쪽. z+ 화면 밖으로 나오는 방향.

3D Coordinate Systems

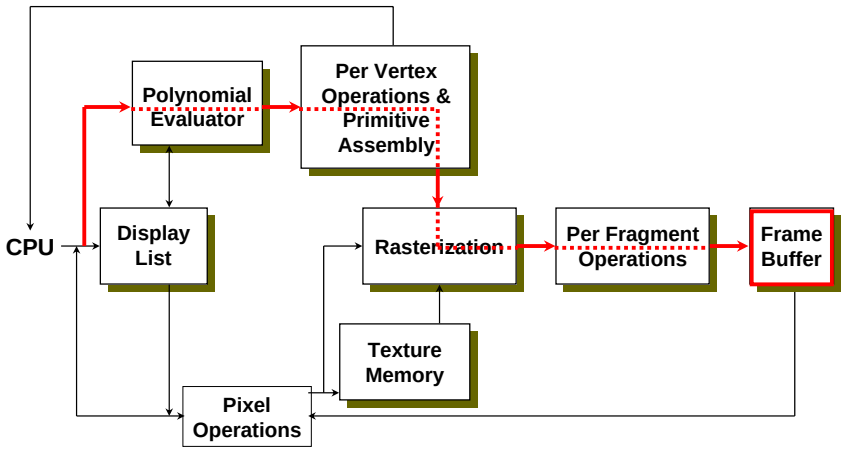


- Direct3D는 왼손 좌표계 (Left-handed coordinate system).
- x+ 오른쪽. y+ 위쪽. z+ 화면 안쪽으로 들어가는 방향.

Graphics Programming

- OpenGL & GLUT API를 사용한 그래픽스 프로그래밍 소개
- OpenGL Programming Guide “red book”
- OpenGL Reference Manual “blue book”
- OpenGL Shading Language “orange book”

OpenGL Architecture



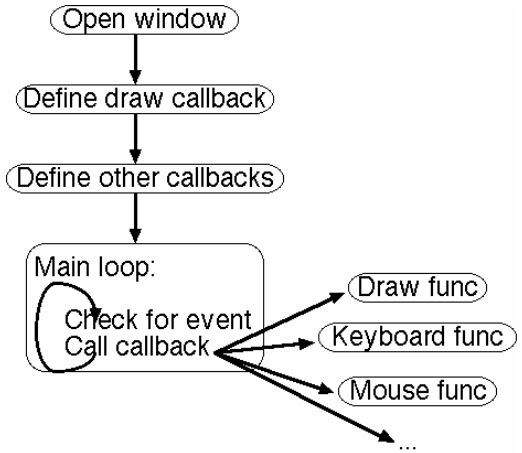
OpenGL

- OpenGL은 그래픽스 파이프라인 (graphics *pipeline*)을 모델로 사용하는 3D graphics library이다.
- OpenGL은 window system에 독립적인 API
- OpenGL은 operating system에 독립적인 API
- OpenGL libraries:
  - GL - graphics
  - GLU - utilities
  - GLX (X window)/ WGL (Windows)/ AGL (Mac) / PGL (IBM) – windowing toolkits
  - GLUT – multi-platform windowing API

GLUT (OpenGL Utility Toolkit)

- Mark J. Kilgard가 개발한 portable windowing and interaction API
  - Prefix "glut"로 시작
  - 대부분의 window system에 보편적인 기능들을 wrapping한 상위 interface제공 (portable across all PC and workstation OS platforms)
  - OpenGL이 제공하는 범위보다 상위 수준의 utility function도 제공
  - UNIX/X-window에서 개발된 code를 그대로 재사용 가능
  - Win32, MFC, Xlib을 알 필요가 없음
  - 그러나 Window system의 기능을 제한적으로만 이용 가능

GLUT Program Structure



GLUT Initialization

- void glutInit(int *argc, char **argv)
  - GLUT와 OpenGL환경을 초기화. 인자(augment)에는 main의 인자를 그대로 건네줌
- void glutInitDisplayMode(unsigned int mode)
  - 화면에 그래픽이 어떻게 그려질지를 지정하는 디스플레이 모드 (display mode)를 설정
  - 인자 mode는 GLUT 기호상수 (symbolic GLUT constant)를 사용
  - GLUT_SINGLE는 싱글버퍼를 사용한다는 의미
  - GLUT_RGB는 색의 지정을 RGB로 사용한다는 의미

GLUT Initialization

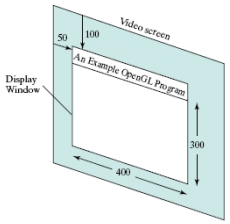
Buffer	GLUT BIT MASK	Purpose
Color buffer	GLUT_RGBA GLUT_RGB GLUT_INDEX	 Use RGB color Use color indexing
	GLUT_ALPHA	
	GLUT_SINGLE GLUT_DOUBLE	Single-buffer Double-buffer
	GLUT_STEREO	
	GLUT_LUMINANCE	
	GLUT_MULTISAMPLING	
Other buffers	GLUT_ACCUM	
	GLUT_DEPTH	Allow depth-buffer
	GLUT_STENCIL	

GLUT Display Mode

GLUT Initialization

- void glutInitWindowPosition(int x, int y)
  - 화면에서 디스플레이될 윈도우의 초기위치 지정
- void glutInitWindowSize(int width, int height)
  - 디스플레이될 윈도우의 크기 지정 (크기는 pixel 단위로 지정)

```
void glutInitWindowPosition(50, 100);
void glutInitWindowSize(400, 300);
```



GLUT Window Management

- int glutCreateWindow(char *name)
  - 윈도우를 여는 함수
  - 인자 name은 윈도우의 이름이 타이틀 바에 표시됨
- int glutCreateSubWindow(int win, int x, int y, int w, int h)
- void glutSetWindow(int win), int glutGetWindow(void)
- void glutDestroyWindow(int win)
- void glutPostRedisplay(void)
  - 윈도우가 새로 그려져야 할 필요가 있는 경우를 표시하는 일을 한다.
- void glutSwapBuffers(void)
  - 더블버퍼(double buffer)일 경우 현재 윈도우의 버퍼를 swap한다.
- void glutPositionWindow(int x, int y)
- void glutReshapeWindow(int w, int h)
  - 현재 윈도우의 크기가 바뀔 때 부른다.

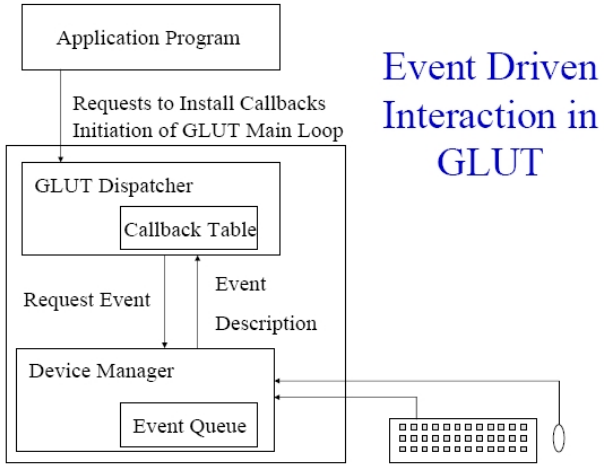
GLUT Window Management

- void glutFullScreen(void)
- void glutPopWindow(void)
- void glutPushWindow(void)
- void glutShowWindow(void), void glutHideWindow(void), void glutIconifyWindow(void)
- void glutSetWindowTitle(char *name), void glutSetIconTitle(char * name)
- void glutSetCursor(int cursor)
  - GLUT_CURSOR_RIGHT_ARROW, GLUT_CURSOR_LEFT_ARROW, GLUT_CURSOR_INFO, GLUT_CURSOR_DESTROY, GLUT_CURSOR_HELP, GLUT_CURSOR_CYCLE, GLUT_CURSOR_SPRAY, GLUT_CURSOR_WAIT, GLUT_CURSOR_TEXT, …

GLUT Main Loop

- void glutMainLoop(void)
  - GLUT 메인 루프. 이 함수의 호출로 프로그램은 이벤트를 기다리는 상태임.
- Main loop
  - Repeat Forever:
    If Empty? (EventQueue)
    Then Invoke function CallbackTable[Idle].
    Else 1. Let Event = Dequeue(EventQueue).
    2. Let Type = Type field of Event.
    3. Let Param = Parameters field of Event.
    4. Invoke function CallbackTable[Type] with Param as parameters.

Event Driven Interaction in GLUT



Event Driven Interaction in GLUT

Event Driven Interaction in GLUT

- 응용 프로그램 (Application Program)
  - GLUT에게 "Callback Table"에 함수를 등록시켜줄 것을 요청
  - GLUT main loop를 시작 (프로그램 종료 시까지 실행됨)
- 장치 관리자 (Device Manager)
  - 마우스나 키보드에서 발생되는 인터럽트(interrupt)에 의해 활성화됨
  - 이벤트큐 (event queue)에 이벤트 정보를 넣는 것으로 인터럽트를 처리함.
- 디스패쳐 반복작업 (GLUT Dispatcher Repeatedly)
  - 장치 관리자로부터 이벤트에 대한 정보를 얻음
  - Callback Table에서부터 발생한 이벤트에 해당되는 함수를 찾아 실행

GLUT Events

- GLUT의 Events는 아래와 같다:
  - Key pressed
  - Key released
  - Mouse button pressed
  - Mouse button released
  - Mouse moved
  - Mouse entered/left window
  - Window resize
  - Window needs redrawing
  - Menu selection
  - Special devices – tablet, space, dial/button box
  - Timer elapsed
  - Idle

GLUT Callbacks

- GLUT defines a basic program structure - an *event loop*, with *callback functions*.
- *callback,* a function that you provide for other code to call when needed; the "other code" is typically in a library
- GLUT uses callbacks for drawing, keyboard & mouse input, and other events.
- Whenever the window must be redrawn, your drawing callback is called.
- Whenever an input event occurs, your corresponding callback is called.

GLUT Callback Registration

- void glutDisplayFunc(void (*func)(void))
  - 현재 실행되는 윈도우를 실제로 보여주는 (즉, 그림을 그리는) callback 함수 포인터를 지정한다.
  - 인자(argument)는 display callback 함수
- void glutOverlayDisplayFunc(void (*func)(void))
- void glutReshapeFunc(void (*func)(int w, int h))
  - 윈도우의 크기가 변할 때 호출하는 callback 함수를 지정한다.
- void glutIdelFunc(void (*func)(void))
  - IDEL인 상태일 경우 호출되는 callback 함수를 지정한다.
- void glutTimerFunc(unsigned int msecs, void (*func)(int value), value)
  - Timer (millisecond단위) 호출

GLUT Callback Registration

- void glutKeyboardFunc(void (*func)(unsigned char key, int x, int y))
  - Keyboard의 ACII-character key가 눌렸을 때 callback
- void glutKeyboardUpFunc
- void glutSpecialFunc(void (*func)(int key, int x, int y))
  - Special keyboard (F1, F2, .. , F12, directional key)가 눌렸을 때 callback
- void glutSpecialUpFunc
- void glutMouseFunc(void (*func)(int button, int state, int x, int y))
  - 마우스 버튼이 눌렸을 때 callback
- void glutMotionFunc(void (*func)(int x, int y))
  void glutPassiveMotionFunc(void (*func)(int x, int y))
  - 마우스 버튼이 눌린 상태에서 마우스가 움직일 때 callback

GLUT Callback Registration

- glutSpaceballMotionFunc
- glutSpaceballRotateFunc
- glutSpaceballButtonFunc
- glutTabletMotionFunc
- glutTabletButtonFunc
- glutMenuStatusFunc
- glutButtonBoxFunc
- glutDialsFunc
- glutEntryFunc
- glutVisibilityFunc

clear.cpp

```
/* minimal program to open & clear a window */
#include <GL/glut.h>
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glFlush();
}
void keyboard(unsigned char key, int x, int y)
{
    switch (key) {
      case 'q':
          exit(0);
          break;
    }
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(200, 200);
    glutInitWindowPosition(0,0);
    glutCreateWindow(argv[0]);
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutMainLoop();
}
```

simple.cpp

```
/* draws a white rectangle on a black background */
#include <GL/glut.h>
void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
    glEnd();
    glFlush();
}
void keyboard(unsigned char key, int x, int y)
{
    /* 변경없음 */
}
int main(int argc, char** argv)
{
    /* 변경없음 */
}
```

Graphics Functions

- 기본 요소 함수 (Primitive functions)
- 속성 함수 (Attribute functions)
- 관측 함수 (Viewing functions) – chap5
- 변환 함수 (Transformation functions) – chap4
- 입력 함수 (Input functions) – GLUT chap3
- 제어 함수 (Control functions) – GLUT
- 질의 함수 (Query functions)

Graphics Functions

- 기본 요소 함수 (Primitive functions)
  - 저수준 객체 (Low-level object)을 정의
  - Points, line segments, polygons, pixels, texts, curves, surfaces
- 속성 함수 (Attribute functions)
  - 기본 요소가 화면상에 나타날 수 있는 방법을 제어
  - Color, line thickness, pattern to fill the polygon, ..
- 관측 함수 (Viewing functions)
  - 합성카메라 (synthetic camera)모델 - 카메라 위치, 방향, 렌즈
  - Viewport, clipping, ..
- 변환 함수 (Transformation functions)
  - 물체의 회전(rotation), 이동(translation), 크기변환 (scaling)

Graphics Functions

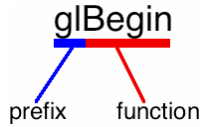
- 입력 함수 (Input functions)
  - 키보드, 마우스, 테블렛 등 입력장치를 다루는 함수
- 제어 함수 (Control functions)
  - 윈도우 시스템과 통신을 가능하게 하고, 프로그램 초기화, 에러처리 등
  - Close, open, resolution setting, mode setting, ..
- 질의 함수 (Query functions)
  - 그래픽스 시스템의 내부적인 상태 (internal states)이나 속성 (properties) 정보

OpenGL Syntax

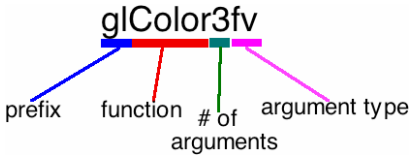
- Basic OpenGL Syntax
  - 함수(function)에 prefix " gl " 사용
    - E.g.: glBegin, glClear, glCopyPixels, glPolygonMode
  - 기호상수(symbolic constant)에 prefix "GL" 사용 & 각각의 단어는 '_'로 연결
    - E.g.: GL_2D, GL_RGB, GL_CCW, GL_POLYGON, GL_AMBIENT_AND_DIFFUSE
  - 자료타입 (data type)에 prefix "GL"
    - E.g. GLbyte, GLshort, GLint, GLfloat, GLdouble, GLboolean

OpenGL Functions

- OpenGL함수 이름은 "gl"로 시작
- 함수 인자 (argument)가 하나인 경우:

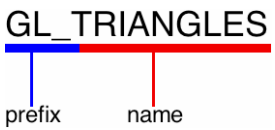


- 함수 인자 (argument)를 여러 개 받는 경우:



OpenGL Constants

- OpenGL의 상수 (Constant)는 함수 인자에 사용
- 상수의 이름은 전부 대문자를 사용하며 "GL"로 시작:



e.g. glBegin()에 들어갈 수 있는 인자들 (arguments)의 예:

- GL_POINTS
- GL_LINES
- GL_TRIANGLES
- GL_POLYGON

OpenGL State

- GL rendering consists of geometry + state
- Both are passed to graphics hardware via function calls
- To draw something, the necessary state attributes (e.g. color) are set first. Then, the geometry (e.g. triangle data) is passed
- State is retained until changed.
- State changes do not affect any geometry already drawn.

OpenGL State

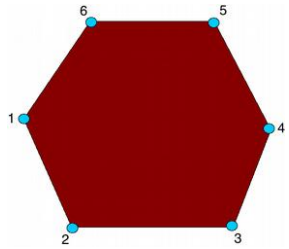
- State:
  - Color
  - Drawing style
  - Material properties
  - Light sources
  - Texture
  - Transformations

simple.cpp

```
#include <GL/glut.h>
void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
        glColor3f(1.0, 0.0, 0.0); glVertex2f(-0.5, -0.5);
        glColor3f(1.0, 1.0, 0.0); glVertex2f(-0.5, 0.5);
        glColor3f(0.0, 1.0, 0.0); glVertex2f(0.5, 0.5);
        glColor3f(0.0, 0.0, 1.0); glVertex3f(0.5, -0.5);
    glEnd();
    glFlush();
}

void keyboard(unsigned char key, int x, int y)
{
    /* 변경없음 */
}
int main(int argc, char** argv)
{
    /* 변경없음 */
}
```

OpenGL Geometry

- 가상의 공간을 구성하는 각 물체를 표현하는데 있어 가장 기본이 되는 요소
- 실시간 그래픽스에서는 주로 가장 단순한 형태의 표현 방법인 linear primitives를 사용
  - Point, vertex
  - Line segments
  - Polygon
  - Polyhedron



OpenGL Geometry

- 기하학적 객체를 정의하기 위해서는:

```
glBegin(…);
glVertex*(...);
…
glEnd();
```

- void glBegin(GLenum mode)

  void glEnd(void)
  - 도형을 그리려면, glBegin()과 glEnd() 사이에 그 도형의 각 정점(vertex)의 좌표치를 설정하는 함수를 둠. mode에 GL_POINTS, GL_LINES, GL_POLYGON 등 도형의 타입을 지정.
- void glVertex*()
  - 이 함수는 2차원의 좌표치를 설정하는 사용. 인수의 형태는 Gldouble임. Float 형태는 glVertex2f(..)를 int 형태는 glVertex2i(..)를 사용함.

OpenGL Geometry

glVertex3fv(*v*)

Number of components

2 - (x,y)
3 - (x,y,z)
4 - (x,y,z,w)

Data Type

b - byte
ub - unsigned byte
s - short
us - unsigned short
i - int
ui - unsigned int
f - float
d - double

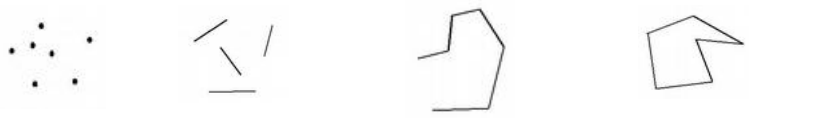
Vector

omit "v" for scalar form

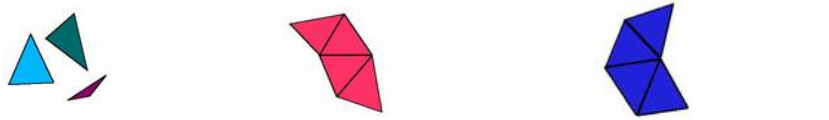
glVertex2f(x, y)

OpenGL Geometry Primitives

GL_POINTS GL_LINES GL_LINE_STRIP GL_LINE_LOOP



GL_TRIANGLES GL_TRIANGLE_STRIP GL_TRIANGLE_FAN



GL_QUADS GL_QUAD_STRIP GL_POLYGON

OpenGL Point Function

```
glBegin(GL_POINTS);
        glVertex2i (50, 100);
        glVertex2i (75, 150);
        glVertex2i (100, 200);
glEnd();
```

```
GLint pt1[2] = {50, 100};
GLint pt2[2] = {75, 150};
GLint pt3[2] = {100, 200};
glBegin(GL_POINTS);
        glVertex2iv(pt1);
        glVertex2iv(pt2);
        glVertex2iv(pt3);
glEnd();
```

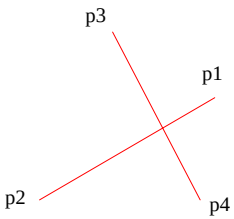
OpenGL Point Function

```
glBegin(GL_POINTS);
        glVertex3f(-78.05, 909.72, 14.60);
        glVertex3f(261.91, -5200.67, 188.33);
glEnd();
```

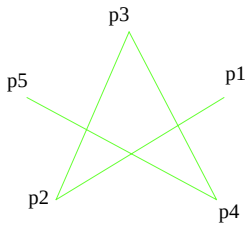
```
class wcPt2D {
public:
        GLfloat x, y;
};
wcPt2D pointPos;

pointPos.x = 120.75;
pointPos.y = 45.30;
glBegin(GL_POINTS);
        glVertex2f(pointPos.x, pointPos.y);
glEnd();
```

OpenGL Line Function

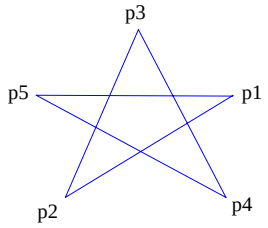
```
glBegin(GL_LINES);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
glEnd();
```




```
glBegin(GL_LINE_STRIP);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
glEnd();
```

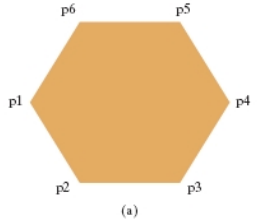



```
glBegin(GL_LINE_LOOP);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
glEnd();
```

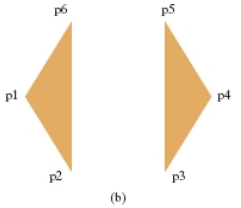


OpenGL Filling Polygon

```
glBegin(GL_POLYGONS);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
    glVertex2iv(p6);
glEnd();
```



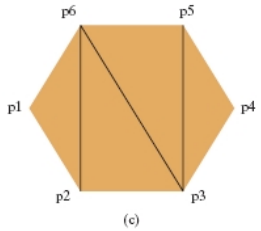

```
glBegin(GL_TRIANGLES);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p6);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
glEnd();
```

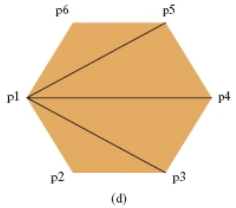


OpenGL Filling Polygon

```
glBegin(GL_TRIANGLE_STRIP);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p6);
    glVertex2iv(p3);
    glVertex2iv(p5);
    glVertex2iv(p4);
glEnd();
```

```
glBegin(GL_TRIANGLE_FAN);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
    glVertex2iv(p6);
glEnd();
```

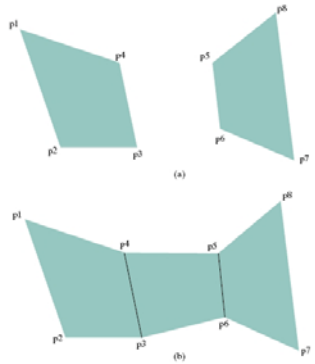




OpenGL Filling Polygon

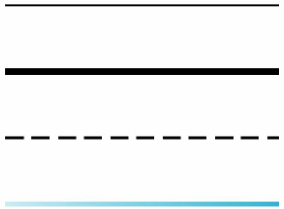
```
glBegin(GL_QUADS);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p3);
    glVertex2iv(p4);
    glVertex2iv(p5);
    glVertex2iv(p6);
    glVertex2iv(p7);
    glVertex2iv(p8);
glEnd();
```

```
glBegin(GL_QUAD_STRIP);
    glVertex2iv(p1);
    glVertex2iv(p2);
    glVertex2iv(p4);
    glVertex2iv(p3);
    glVertex2iv(p5);
    glVertex2iv(p6);
    glVertex2iv(p8);
    glVertex2iv(p7);
glEnd();
```

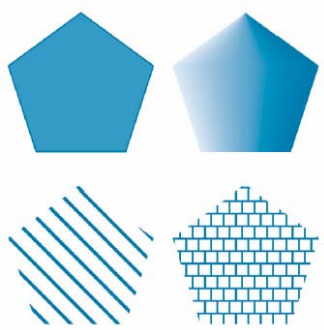


OpenGL Attributes

- 각 기하학적 기본요소 (geometry primitive)는 속성을 갖고 있다. 속성은 기본 요소가 화면상에 나타날 수 있는 방법을 제어한다.
  - Color
  - Line thickness
  - Line styles
  - Polygon patterns



선의 두께나 스타일



다각형 표시방법

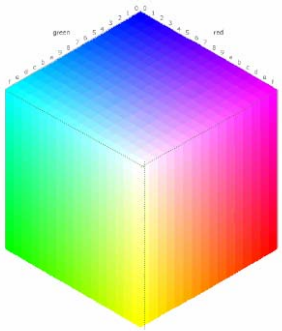
OpenGL Attributes

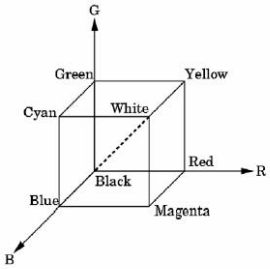
- Vertex
  - 점의 크기 - glPointSize(GLfloat size)
- Line
  - 선분의 폭 – glLineWidth(GLfloat width)
  - 점선의 스타일 – glLineStipple(GLint factor, GLushort pattern)
- Polygon
  - 외형과 사용면 지정 – glPolygonMode(GLenum face, GLenum mode)
    - Face: GL_FRONT_AND_BACK, GL_FRONT, GL_BACK
    - Mode: GL_POINT, GL_LINE, GL_FILL
  - 앞면 지정 – glFrontFace(GLenum mode)
    - Mode: GL_CCW, GL_CW
  - 스티플링 – glPolygonStipple(const GLubyte *mask)
    - Mast: 32x32 비트 윈도우 정렬 스티플 패턴

OpenGL Attributes

- OpenGL Color Model
  - RGB (Red Green Blue) or RGBA(Red Green Blue Alpha)
  - RGB 색이 따로 분리돼서 framebuffer에 저장되어 있음.

glColor3f(1.0, 0.0, 0.0); //색값은 0.0 ~ 1.0
glColor3ub(255, 0.0, 0.0); //색값은 0~255

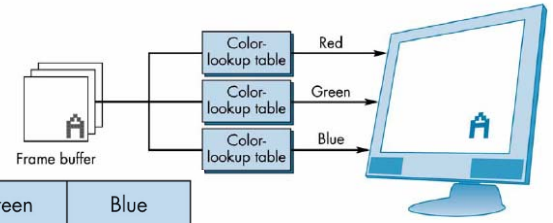




OpenGL Attributes

- OpenGL Color Model
  - Color Index는 메모리를 적게 사용함. 현재는 크게 쓰이지 않음.

glIndexi(element);



Input	Red	Green	Blue
0	0	0	0
1	$2^m - 1$	0	0
.	0	$2^m - 1$	0
.	.	.	.
.	.	.	.
$2^k - 1$	.	.	.

m bits *m* bits *m* bits