CSC 307 1.0
Graphics Programming

Budditha Hettige
Department of Statistics and Computer Science

02

Graphics Programming

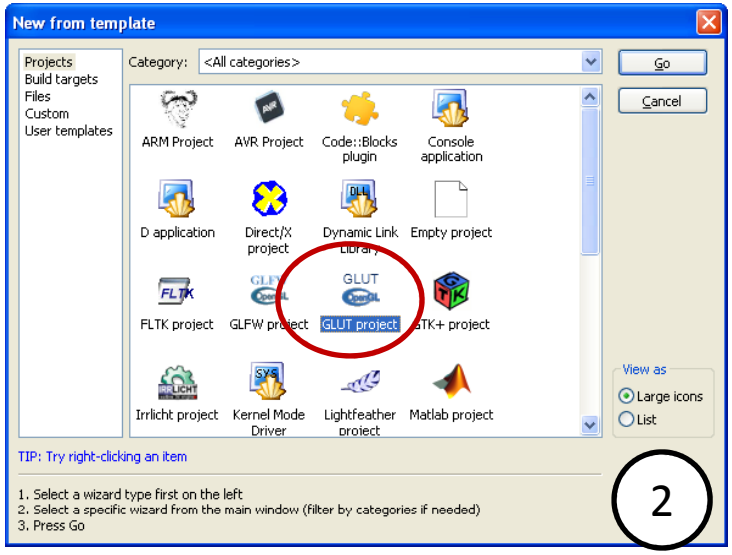
OpenGL & GLUT in Code::Blocks

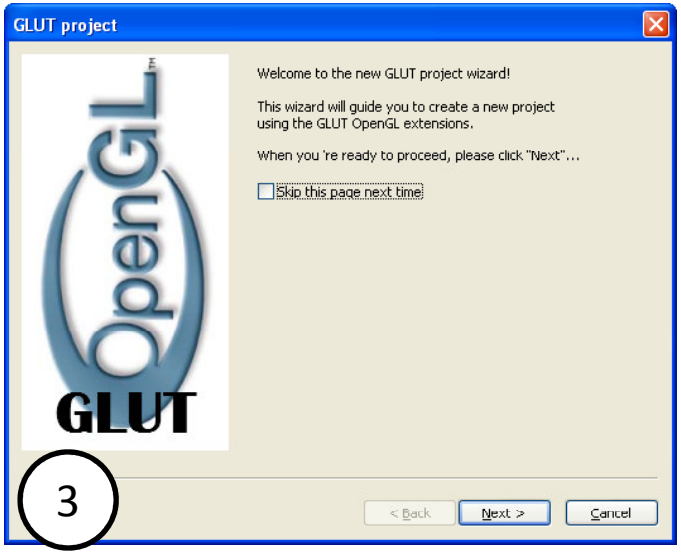
OpenGL & Code::block

- Download Code::Blocks
  - http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/
- Download the GLUT bin file from
  - http://www.xmission.com/~nate/glut.html
- Save files as
  - Copy glut32.dll to
    - **C:\windows\system**
  - Copy glut32.lib to
    - **C:\Program Files\CodeBlocks\MinGW\lib**,
  - Copy glut.h to
    - **C:\Program Files\CodeBlocks\MinGW\include\GL**.

Code::Bolock Project

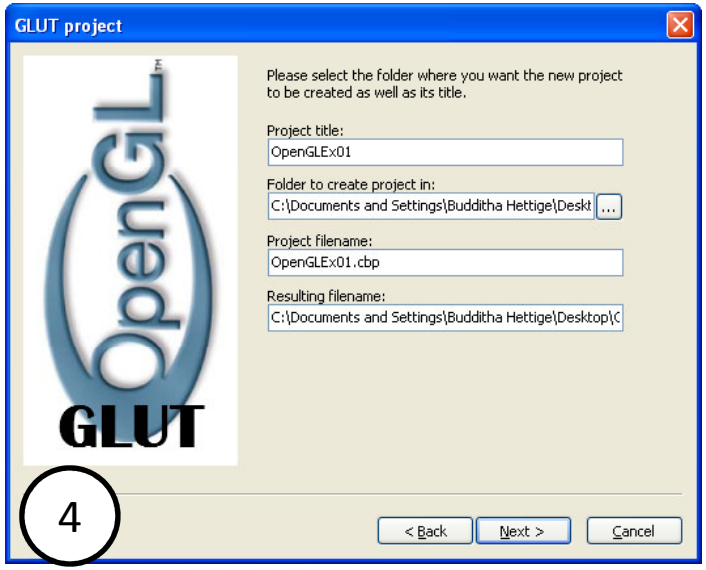
1. Start Code::Blocks and make a new project.
2. Select to make a new **GLUT project** and press **Go** to continue.
3. Press **Next** at this menu

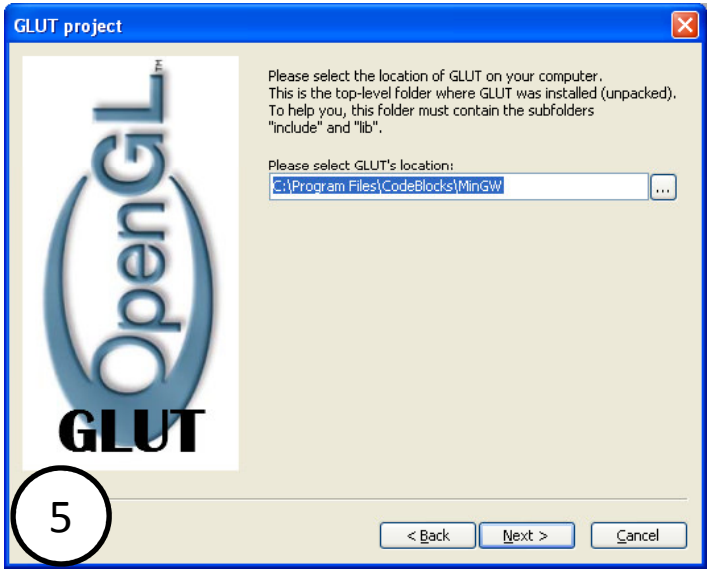




Code::Bolock Project

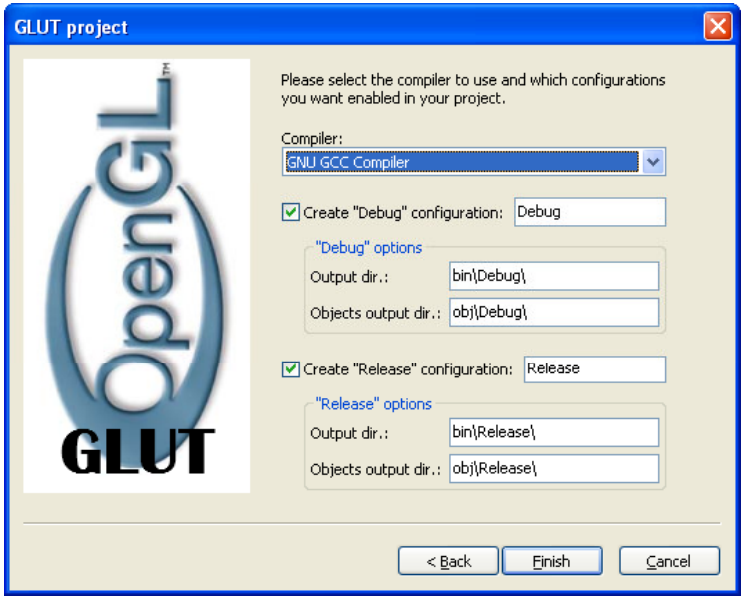
4. Give a **project title**, and a **location** where to create the project and then press **Next**
5. Tell Code::Blocks to where you stored your GL files, then press **Next**

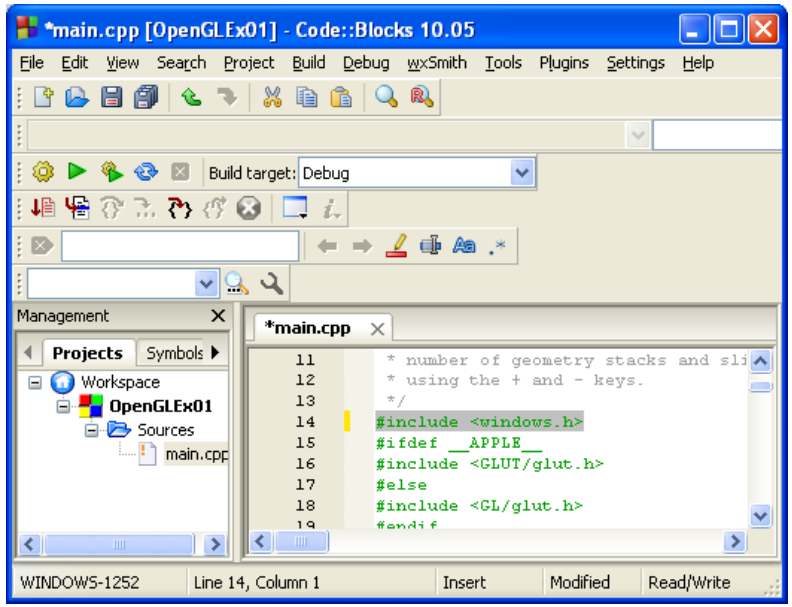




Code::Bolock Project

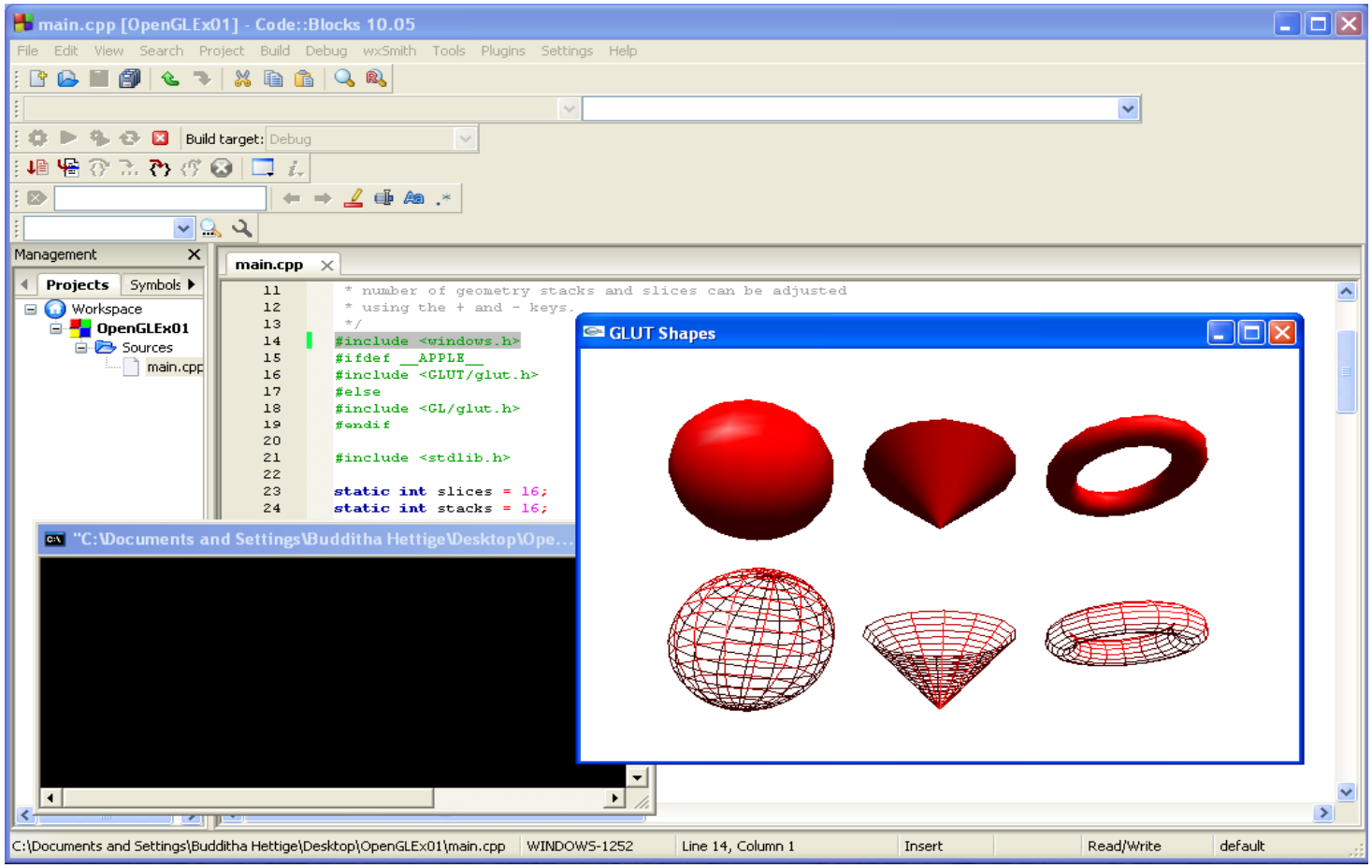
- *Set compiler as "GNU GCC Compiler"*, and press **Finish**.
- Open up the sample source file by double clicking on it
- Add **#include <windows.h>** at line 14





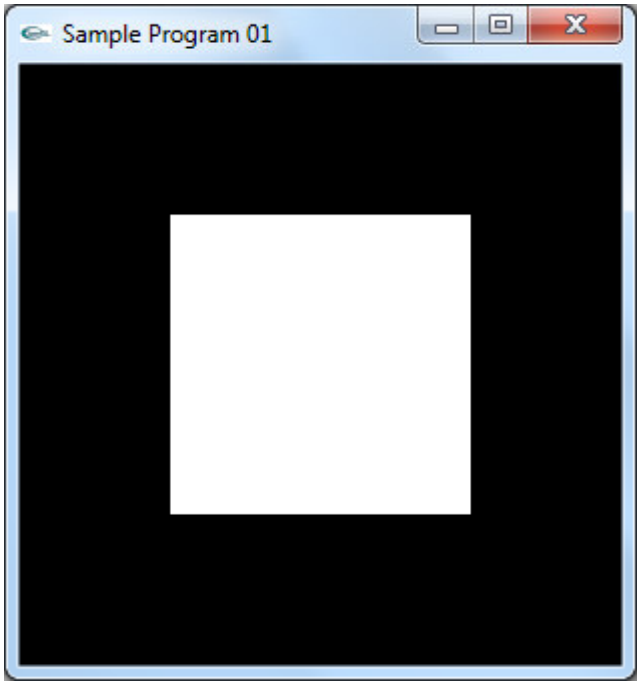
Code::Bolock Project

- Compile and build an application



Sample 01

```
# include <windows.h>
#include <GL/glut.h>
void mydisplay()
{
    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();
}
int main(int argc, char** argv)
{
     glutInit(&argc,argv);
     glutCreateWindow("simple");
     glutDisplayFunc(mydisplay);
     glutMainLoop();
}
```



GLUT Functions

```
glutInit(int *argc, char** argv);
```
Initializes a window session.

```
glutCreateWindow(char *name);
```
Creates a window with title *name.

```
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
```
Sets the display mode to single buffered and RGB color.

```
glutInitWindowSize (GLsizei h, GLsizei w);
```
Sets initial window size to h x w.

```
glutInitWindowPosition(x,y);
```
Sets initial window position to (x, y).

GLUT Functions

- void glFlush()

  force execution of GL commands in finite time
- void glutDisplayFunc(void (*func)(void));

  sets the display callback for the *current window*.
- void glutMainLoop(void);

  Enters the GLUT event processing loop

OpenGL Attributes

- `glClearColor(1.0, 1.0, 1.0, 0.0);`
  - Sets background color to white
  - Fourth argument is transparency; 0.0 is opaque
  - Sets a state variable
- `glPointSize(2.0);`
  - Sets point size to be 2 pixels wide
  - Note that this is not a device-independent attribute
  - Sets a state variable

glClear

- **Clearing the Color Buffer**
  - glClear(GL_COLOR_BUFFER_BIT);
- Values
  - GL_COLOR_BUFFER_BIT

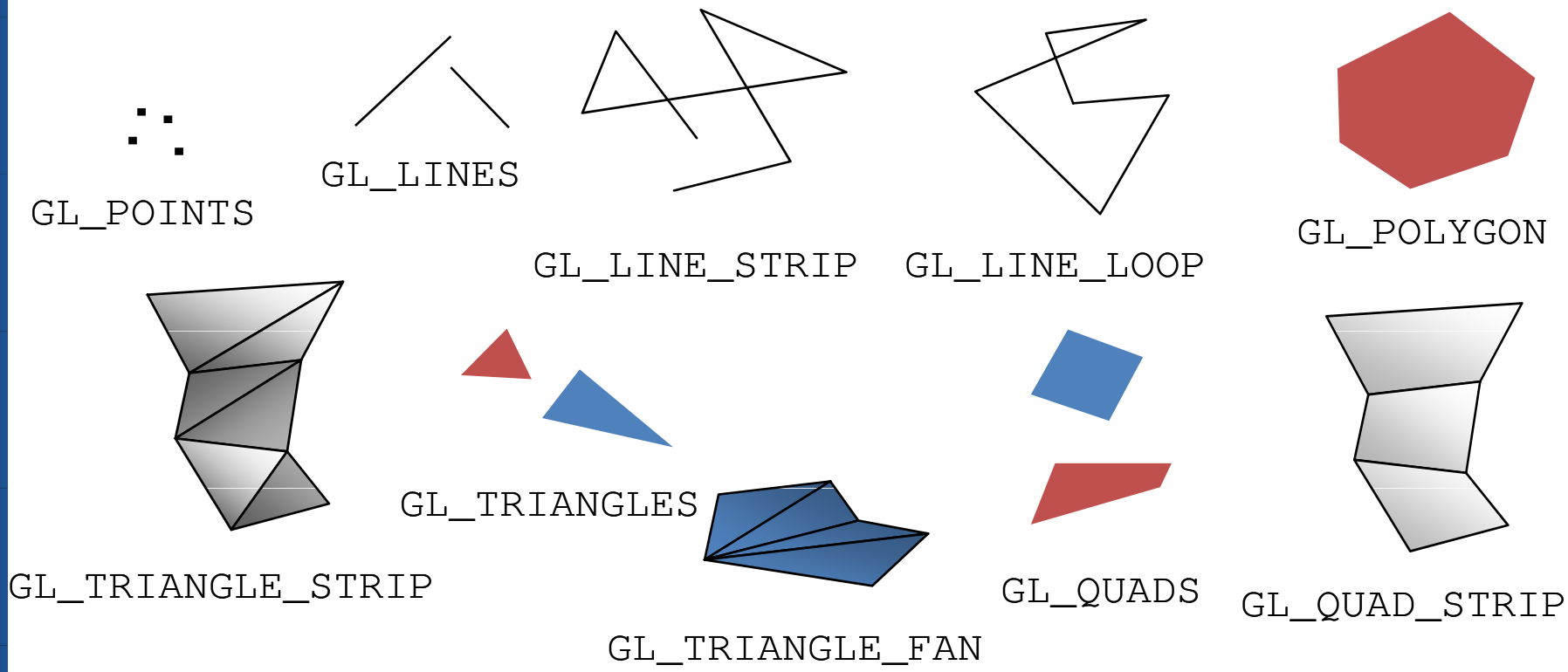
    Indicates the buffers currently enabled for color writing.
  - GL_DEPTH_BUFFER_BIT

    Indicates the depth buffer.
  - GL_ACCUM_BUFFER_BIT

    Indicates the accumulation buffer.
  - GL_STENCIL_BUFFER_BIT

    Indicates the stencil buffer.

OpenGL Geometric Primitives



OpenGL Primitive Syntax

```
glBegin ( type );
glVertex* ( );
    .
    .
    .
    .

glVertex* (  );
glEnd ( );
```

```
glBegin(GL_POLYGON);
        glVertex2f(-0.5, -0.5);
        glVertex2f(-0.5, 0.5);
        glVertex2f(0.5, 0.5);
        glVertex2f(0.5, -0.5);
glEnd();
```

```
glBegin(GL_TRIANGLES);
        glVertex3f( 0.0f, 1.0f, -10.0f);
        glVertex3f(-1.0f,-1.0f, -10.0f);
        glVertex3f( 1.0f,-1.0f, -10.0f);
glEnd();

glBegin(GL_LINES);
        glVertex3f(0.25, 0.25, 0.0);
        glVertex3f(0.75, 0.75, 0.0);
glEnd();
```

Sample 02

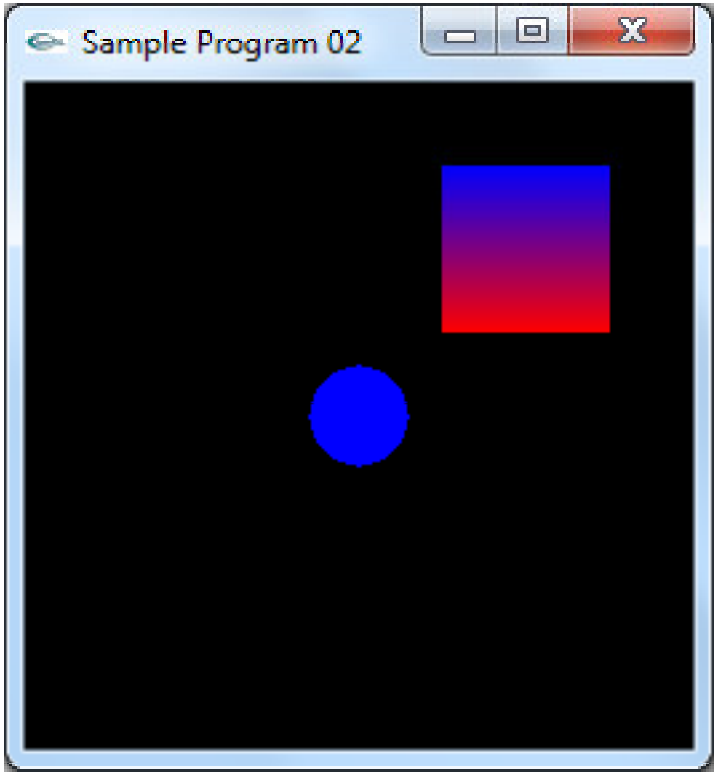
```
# include <windows.h>
# include <GL/glut.h>

void display(void)
{
glClear (GL_COLOR_BUFFER_BIT);
glColor3f (1.0, 0.0, 0.0); //red
   glBegin(GL_QUADS);
      glVertex3f (0.25, 0.25, 0.0);
      glVertex3f (0.75, 0.25, 0.0);
      glColor3f (0.0, 0.0, 1.0); //blue
      glVertex3f (0.75, 0.75, 0.0);
      glVertex3f (0.25, 0.75, 0.0);
   glEnd();
   glutSolidSphere(0.15,12,2);  //draw a sphere
  glFlush ();
}
```

```
void init (void)
{
   glClearColor (0.0, 0.0, 0.0, 0.0);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);
}
```

Sample 02

```
int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitDisplayMode (GLUT_SINGLE |
     GLUT_RGB);
   glutInitWindowSize (250, 250);
   glutInitWindowPosition (100, 100);
   glutCreateWindow ("Sample Program 02");
   init ();
   glutDisplayFunc(display);
   glutMainLoop();
}
```



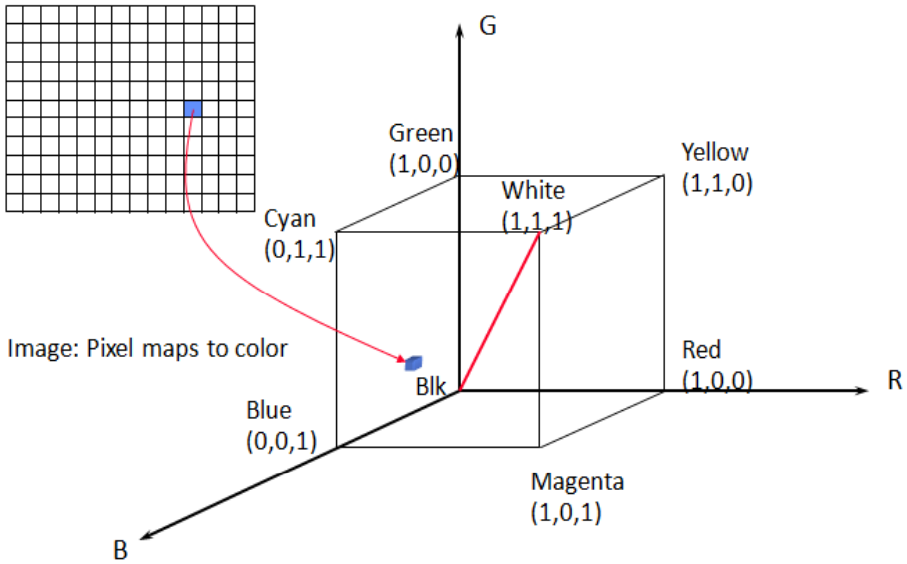
glutInitDisplayMode

- Sets the *initial display mode*.
  - glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
- Values
  - GLUT_RGBA
  - GLUT_RGB
  - GLUT_INDEX
  - GLUT_SINGLE
  - GLUT_DOUBLE
  - GLUT_ACCUM
  - GLUT_ALPHA
  - GLUT_DEPTH
  - GLUT_STENCIL
  - GLUT_MULTISAMPLE
  - GLUT_STEREO
  - GLUT_LUMINANCE

glColor

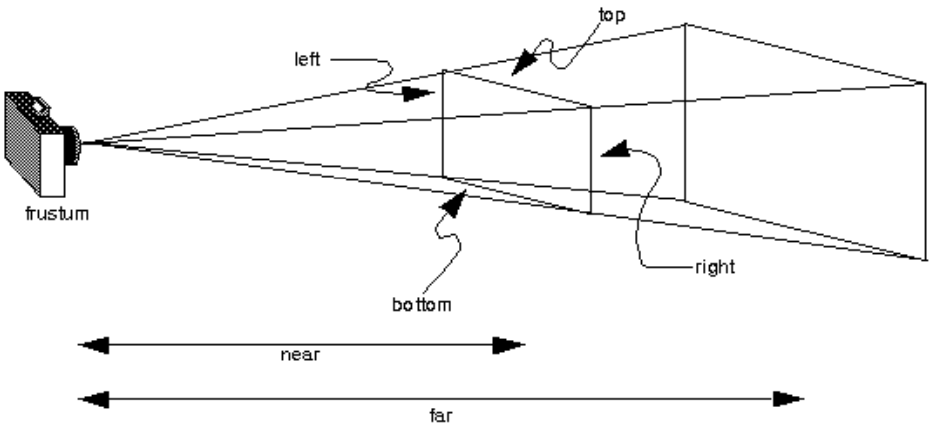
- Set the current color
  - glColor3f (1.0, 0.0, 0.0);
- Example
  - void glColor3i(GLint red, GLint green, GLint blue);
  - void glColor3f(GLfloat red, GLfloat green, GLfloat blue);

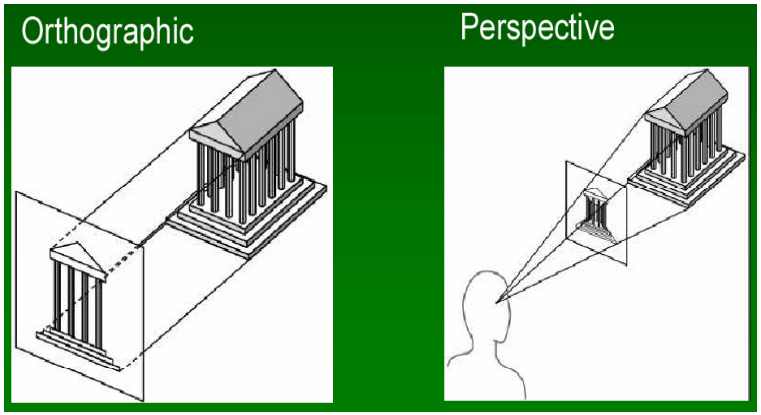
  - glColor3f (1.0, 0.0, 0.0); //red
  - glColor3f (0.0, 0.0, 1.0); //blue



Projection Transformation

- Transformation from scene to image
- Orthographic projection
  - glOrtho (left, right, bottom, top, near, far)
  - glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);
- Perspective projection
  - glFrustum (left, right, bottom, top, near, far)





OpenGL Transformations

- Before applying modeling or viewing transformations, need to set
  ```
  glMatrixMode(GL_MODELVIEW)
  ```
- Before applying projection transformations, need to set
  ```
  glMatrixMode(GL_Projection)
  ```
- Replacement by either following commands
  ```
  glLoadIdentity();
  glLoadMatrix(M);
  ```
- Multiple transformations (either in modeling or viewing) are applied in **reverse** order

Setting Viewing Matrix

```
glMatrixMode(GL_PROJECTION);
```

Sets the switch so that loaded matrix goes into the projection stack.

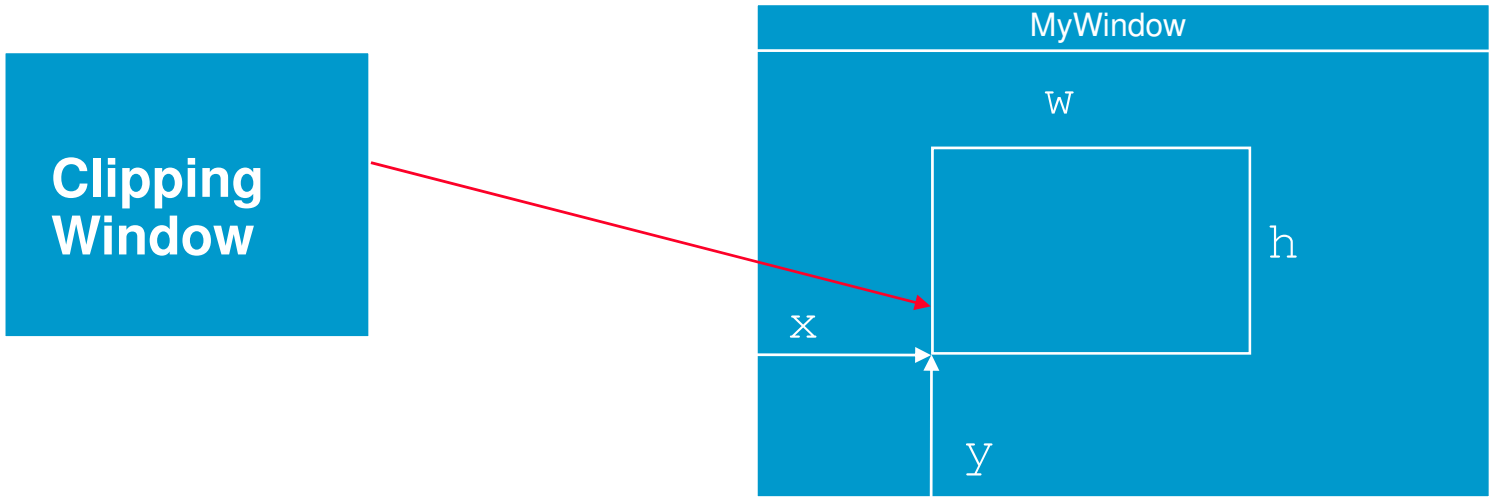
```
glLoadIdentity();
```

Pushes an identity matrix onto the stack;

```
gluOrtho2D(GLdouble left, Gldouble right,
           Gldouble bottom, Gldouble top);
```

Sets the current view to an orthographic projection with view volume bounded by x = left, x = right, y = bottom, y = top, z = -1.0 and z = 1.0.

Viewport Transformation

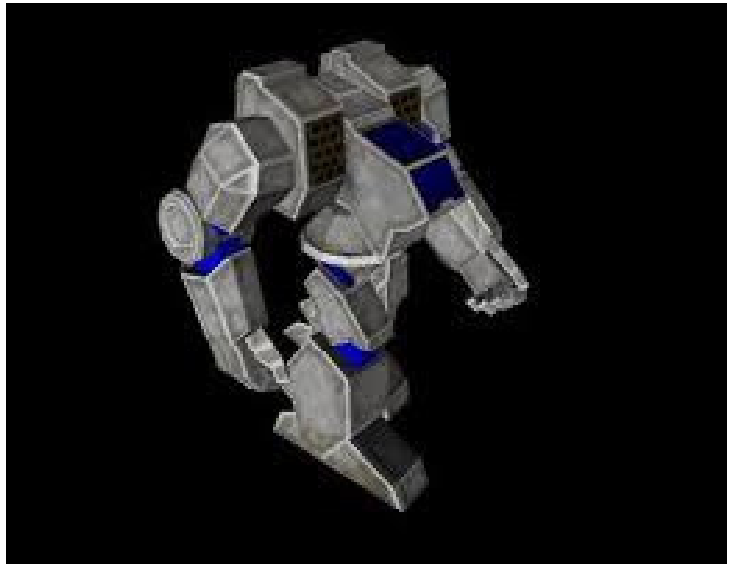



```
void glViewport(Glint x, GLint y, GLsizei w, Glsizei h);
```

Default viewport corresponds to entire window drawable area.

Primitives

- GL_POINTS
- GL_LINES
- GL_TRIANGLES
- GL_TRIANGLE_STRIP
- GL_QUAD_STRIP
- GL_LINE_STRIP
- GL_LINE_LOOP
- GL_QUADS
- GL_POLYGON
- GL_TRIANGLE_FAN



OpenGL Applications