The GL Utility Toolkit (GLUT)

The GL Utility Toolkit (GLUT) serves two major purposes:
1. It interfaces with your operating system and window system.
2. It provides various application utilities, such as drawing 3D shapes for you.

You can find GLUT (actually freeGLUT) at:
http://freeglut.sourceforge.net/

What is GLUT?
You don't actually have to go out here. We will give you some libraries that are ready-to-use.

Using GLUT to Setup the Window

All the GLUT_XXX constants are #defined in glut.h:

- GLUT_RGBA         I want to display colors
- GLUT_DOUBLE    I want to do double-buffering
- GLUT_DEPTH       I want to use a depth-buffer while rendering

glutInitDisplayMode( GLUT_RGBA | GLUT_DOUBLE | GLUT_DEPTH );

// set the initial window configuration:

glutInitWindowPosition( 0, 0 );

// open the window and set its title:

MainWindow = glutCreateWindow( WINDOWTITLE );

glutSetWindowTitle( WINDOWTITLE );

Constants not beginning with GL_ or GLUT_ are user-defined

Using GLUT to Specify Event-driven Callback Functions

For example, the Keyboard( ) function gets called whenever a keyboard key is hit:

void Keyboard( unsigned char c,   int x, int y )
{

  if( DebugOn != 0 )
    fprintf( stderr, "Keyboard: '%c' (0x%0x)\n", c, c );

  switch( c )
  {
    case 'o':  case 'O':
      WhichProjection = ORTHO;
      break;
    case 'p':  case 'P':
      WhichProjection = PERSP;
      break;
    case 'q':  case 'Q':
      case ESCAPE:
      DoMainMenu( QUIT );     // will not ever return
      break;                              // keep the compiler happy
    default:
      fprintf( stderr, "Don't know what to do with keyboard hit: '%c' (0x%0x)\n", c, c );

  // force a call to Display( ):
  glutSetWindow( MainWindow );
  glutPostRedisplay( );
}

The Keyboard Callback Function

The MouseButton Callback Function

Where the mouse was when the button was hit

GOOD PRACTICE

Assign new display parameter values depending on what key was hit

Good programming practice

glutPostRedisplay() forces your Display() function to be called to re-draw the scene with the new display parameter values

glutPostRedisplay();

Where the mouse was when the key was hit

Where the mouse was when the button was hit

GLUT_DOWN or GLUT_UP

Which button was hit

GOOD PRACTICE
The MouseMotion Callback Function

When the mouse moved to

- if the mouse moved with the left button down, do a rotate
- if the mouse moved with the right button down, do a scale
- if the mouse moved with the middle button down, do a translate

glutPostRedisplay() forces your Display() function to be called to redraw the scene with the new display parameter values.

The Animate Idle Callback Function

The Idle Function gets called when the GLUT event handler has nothing else to do

glutIdleFunc( Animate );

Animate( )

Setting it up in InitGraphics( )

We’ll talk about this later. This is a good way to control your animation!

The GLUT 3D Objects

- glutSolidSphere( radius, slices, stacks )
- glutWireSphere( radius, slices, stacks )
- glutSolidCube( size )
- glutWireCube( size )
- glutSolidTorus( innerRadius, outerRadius, nsides, nrings )
- glutWireTorus( innerRadius, outerRadius, nsides, nrings )
- glutSolidTetrahedron( )
- glutWireTetrahedron( )
- glutSolidOctahedron( )
- glutWireOctahedron( )
- glutSolidIcosahedron( )
- glutWireIcosahedron( )
- glutSolidDodecahedron( )
- glutWireDodecahedron( )
- glutSolidCone( base, height, slices, stacks )
- glutWireCone( base, height, slices, stacks )
- glutSolidTeapot( size )
- glutWireTeapot( size )

In case you have a hard time remembering which function does which, here’s a quick reference: