```
 1: // $Id: circles.cpp,v 1.44 2016-04-28 16:11:53-07 - - $
 2:
 3: // Draw several ellipses in window.
 4:
 5: #include <algorithm>
 6: #include <cmath>
 7: #include <iostream>
 8: #include <string>
 9: #include <unordered_map>
10: using namespace std;
11:
12: #include <GL/freeglut.h>
13: #include <libgen.h>
14:
15: // Characteristics of the window.
16: struct {
17:    string name;
18:    int width {512};
19:    int height {384};
20: } window;
21:
22: struct rgbcolor { GLubyte ubv[3]; };
23: const unordered_map<string,rgbcolor> colors {
24:    {"red",     {0xFF, 0x00, 0x00}},
25:    {"green",   {0x00, 0xFF, 0x00}},
26:    {"blue",    {0x00, 0x00, 0xFF}},
27:    {"cyan",    {0x00, 0xFF, 0xFF}},
28:    {"magenta", {0xFF, 0x00, 0xFF}},
29:    {"yellow",  {0xFF, 0xFF, 0x00}},
30:    {"white",   {0xFF, 0xFF, 0xFF}},
31:    {"black",   {0x00, 0x00, 0x00}},
32: };
33:
34: void draw_xy_graph (const rgbcolor& color) {
35:    glLineWidth (1);
36:    glBegin (GL_LINES);
37:    glColor3ubv (color.ubv);
38:    glVertex2f (-window.width / 2, 0);
39:    glVertex2f (+window.width / 2, 0);
40:    glVertex2f (0, -window.height);
41:    glVertex2f (0, +window.height);
42:    glEnd();
43: }
44:
45: void draw_circle (const rgbcolor& color, size_t multiplier,
46:                   GLfloat radius) {
47:    glLineWidth (2);
48:    glBegin (GL_LINE_LOOP);
49:    glColor3ubv (color.ubv);
50:    const size_t points = multiplier * 4;
51:    const GLfloat theta = 2.0 * M_PI / points;
52:    for (size_t point = 0; point < points; ++point) {
53:       GLfloat angle = point * theta;
54:       GLfloat xpos = radius * cos (angle);
55:       GLfloat ypos = radius * sin (angle);
56:       glVertex2f (xpos, ypos);
57:    }
58:    glEnd();
```

```
 59: }
 60:
 61: // Called by glutMainLoop to display window contents.
 62: void display() {
 63:    cout << __func__ << "()" << endl;
 64:    glClearColor (0.25, 0.25, 0.25, 1.0);
 65:    glClear (GL_COLOR_BUFFER_BIT);
 66:    draw_xy_graph (colors.at("cyan"));
 67:    const GLfloat radius = min (window.width, window.height) / 20.0;
 68:    for (size_t count = 1; count <= 10; ++count) {
 69:       draw_circle (colors.at("green"), count, radius * count);
 70:    }
 71:    glutSwapBuffers();
 72: }
 73:
 74: void reshape (int width, int height) {
 75:    cout << __func__ << "(" << width << "," << height << ")" << endl;
 76:    window.width = width;
 77:    window.height = height;
 78:    glMatrixMode (GL_PROJECTION);
 79:    glLoadIdentity();
 80:    gluOrtho2D (-window.width / 2, +window.width / 2,
 81:                -window.height / 2, +window.height / 2);
 82:    glMatrixMode (GL_MODELVIEW);
 83:    glViewport (0, 0, window.width, window.height);
 84:    glutPostRedisplay();
 85: }
 86:
 87: void close() {
 88:    cout << __func__ << "()" << endl;
 89: }
 90:
 91: void entry (int state) {
 92:    cout << __func__ << "(";
 93:    switch (state) {
 94:       case GLUT_LEFT: cout << "GLUT_LEFT"; break;
 95:       case GLUT_ENTERED: cout << "GLUT_ENTERED"; break;
 96:       default: cout << state; break;
 97:    }
 98:    cout << ")" << endl;
 99: }
100:
101: int main (int argc, char** argv) {
102:    window.name = basename (argv[0]);
103:    glutInit (&argc, argv);
104:    glutInitDisplayMode (GLUT_RGBA | GLUT_DOUBLE);
105:    glutInitWindowSize (window.width, window.height);
106:    glutInitWindowPosition (128, 128);
107:    glutCreateWindow (window.name.c_str());
108:    glutDisplayFunc (display);
109:    glutReshapeFunc (reshape);
110:    glutEntryFunc (entry);
111:    glutCloseFunc (close);
112:    glutMainLoop();
113:    return 0;
114: }
115:
116: //TEST// mkpspdf hello-gl.ps hello-gl.cpp*
```