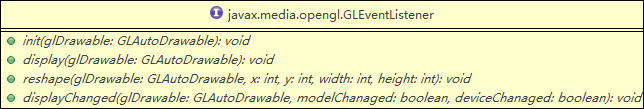
# 创建一个类，实现GLEventListener接口



图表 1 GLEventListener接口

GLEventListener一共有四个方法，int、display、reshape三个方法分别对应OpenGL渲染框架里的”init”、”display”、”reshape”，所以，用户只需要定义一个类，实现这三个方法，就可以实现一个最基本的渲染框架了。

代码清单 1便是定义了一个Renderer类，对GLEventListener的以上三个方法都做了简单的实现。

代码清单 1 实现GLEventlistener接口

|  |
| --- |
| **class** Renderer **implements** GLEventListener {  **private** GLU glu = **new** GLU();  **public** **void** display(GLAutoDrawable gLDrawable) {  **final** GL gl = gLDrawable.getGL();  gl.glClear(GL.*GL\_COLOR\_BUFFER\_BIT* | GL.*GL\_DEPTH\_BUFFER\_BIT*);  gl.glLoadIdentity();  gl.glTranslatef(-1.5f, 0.0f, -6.0f);  gl.glBegin(GL.*GL\_TRIANGLES*); // Drawing Using Triangles  gl.glVertex3f(0.0f, 1.0f, 0.0f); // Top  gl.glVertex3f(-1.0f, -1.0f, 0.0f); // Bottom Left  gl.glVertex3f(1.0f, -1.0f, 0.0f); // Bottom Right  gl.glEnd(); // Finished Drawing The Triangle  gl.glTranslatef(3.0f, 0.0f, 0.0f);  gl.glBegin(GL.*GL\_QUADS*); // Draw A Quad  gl.glVertex3f(-1.0f, 1.0f, 0.0f); // Top Left  gl.glVertex3f(1.0f, 1.0f, 0.0f); // Top Right  gl.glVertex3f(1.0f, -1.0f, 0.0f); // Bottom Right  gl.glVertex3f(-1.0f, -1.0f, 0.0f); // Bottom Left  gl.glEnd(); // Done Drawing The Quad  gl.glFlush();  }  **public** **void** displayChanged(GLAutoDrawable gLDrawable, **boolean** modeChanged, **boolean** deviceChanged) {}  **public** **void** init(GLAutoDrawable gLDrawable) {  GL gl = gLDrawable.getGL();  gl.glClearColor(0.0f, 0.0f, 0.0f, 0.0f);  gl.glShadeModel(GL.*GL\_FLAT*);  }  **public** **void** reshape(GLAutoDrawable gLDrawable, **int** x, **int** y, **int** width, **int** height) {  **final** GL gl = gLDrawable.getGL();  **if** (height <= 0) // avoid a divide by zero error!  height = 1;  **final** **float** h = (**float**) width / (**float**) height;  gl.glViewport(0, 0, width, height);  gl.glMatrixMode(GL.*GL\_PROJECTION*);  gl.glLoadIdentity();  glu.gluPerspective(45.0f, h, 1.0, 20.0);  gl.glMatrixMode(GL.*GL\_MODELVIEW*);  gl.glLoadIdentity();  }  } |

# 将GLEventListener的实例与一个GLCanvas实例绑定

代码清单 2 GLEventListener与GLCanvas绑定

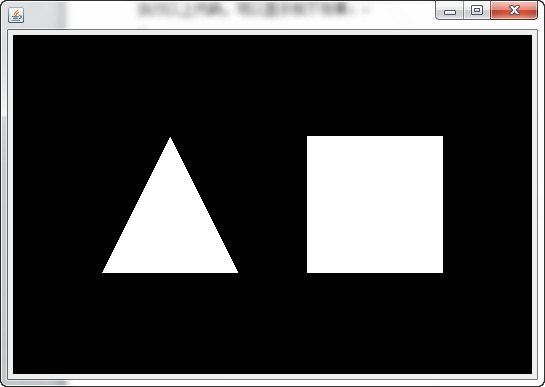
|  |
| --- |
| GLEventListener glListener = **new** Renderer();  GLCanvas glCanvas = **new** GLCanvas(**new** GLCapabilities());  glCanvas.addGLEventListener(glListener); |

# 将GLCanvas嵌入界面

代码清单 3 GLCanvas嵌入JPanel

|  |
| --- |
| **public** **class** AppFrame **extends** JFrame {  **private** JPanel contentPane;  **public** **static** **void** main(String[] args) {  EventQueue.*invokeLater*(**new** Runnable() {  **public** **void** run() {  **try** {  GLCanvas glCanvas = *createGLCanvas*();  AppFrame frame = **new** AppFrame(glCanvas);  frame.setVisible(**true**);  } **catch** (Exception e) {  e.printStackTrace();  }  }  });  }  **public** **static** GLCanvas createGLCanvas() {  GLEventListener glListener = **new** Renderer();  GLCanvas glCanvas = **new** GLCanvas(**new** GLCapabilities());  glCanvas.addGLEventListener(glListener);  **return** glCanvas;  }  **public** AppFrame(GLCanvas glCanvas) {  setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  setBounds(100, 100, 450, 300);  contentPane = **new** JPanel();  contentPane.setBorder(**new** EmptyBorder(5, 5, 5, 5));  contentPane.setLayout(**new** BorderLayout(0, 0));  contentPane.add(glCanvas, BorderLayout.*CENTER*);    setContentPane(contentPane);  }  } |

执行以上代码，可以显示如下效果：



图表 2 使用JOGL渲染基本图形