the center and drawing a circle with the sphere's radius. Listing 17.3 uses these simple projections to visualize the structure of the methane CH_4 molecule.

Listing 17.3 The `Methane` class implements a visualization of the methane molecule CH_4.

```
package org.opensourcephysics.sip.ch17;
import java.awt.*;
import org.opensourcephysics.display.*;

public class Methane implements Drawable {
   static final double cos30 = Math.cos(Math.PI/6);
   static final double sin30 = Math.sin(Math.PI/6);
   // trapezoid height
   static final double h = Math.sqrt(1.0-4.0*cos30*cos30/9.0);
   double[][] atoms = new double[5][];
   Circle circle = new Circle();

   public Methane() {
      // atom locations in 3D homogeneous coordinates
      // C atom at origin
      atoms[0] = new double[] {0, 0, 0, 1};
      // H atom on z-axis
      atoms[1] = new double[] {0, 0, 0.75*h, 1};
      // H atom
      atoms[2] = new double[] {2.0*cos30/3.0, 0, -0.25*h, 1};
      // H atom
      atoms[3] = new double[] {-cos30/3.0, sin30, -0.25*h, 1};
      // H atom
      atoms[4] = new double[] {-cos30/3.0, -sin30, -0.25*h, 1};
   }

   void transform(Rotation3D t) {
      for(int i = 0, n = atoms.length;i<n;i++) {
         t.direct(atoms[i]);
      }
   }

   public void draw(DrawingPanel panel, Graphics g) {
      g.setColor(Color.black);
      int x0 = panel.xToPix(0);
      int y0 = panel.yToPix(0);
      for(int i = 0, n = atoms.length;i<n;i++) {
         int xpix = panel.xToPix(atoms[i][0]);
         int ypix = panel.yToPix(atoms[i][1]);
         g.drawLine(x0, y0, xpix, ypix);
      }
      for(int i = 0, n = atoms.length;i<n;i++) {
         circle.setXY(atoms[i][0], atoms[i][1]);
         circle.draw(panel, g);
      }
   }
}
```

The carbon and hydrogen atom coordinates are given in the `Methane` constructor. These coordinates are stored in a multidimensional array so that we can loop over the atoms in

the `draw` method. The carbon atom (the center of symmetry) is placed at the origin and a hydrogen atom is placed above it along the z-axis. The three remaining hydrogen atoms are placed so as to form a tetrahedron with an H-H separation equal to unity. This orientation can be changed using the `transform` method to rotate the coordinates. Note that the `draw` method draws lines from the origin to each hydrogen atom's coordinates and then draws circles at these coordinates.

Exercise 17.6 Methane

Determine the angle between two hydrogen bonds using the data in the `Methane` class. ■

The `MethaneApp` program uses the `Rotation3D` and `Methane` classes by rotating the methane molecule about the origin in response to mouse actions. The `handleMouseAction` method stores the current mouse position when the mouse is clicked. A `Rotation3D` object is created when the mouse is dragged; the drag distance determines the angle of rotation. If the mouse is dragged vertically, the molecule is rotated about the y-axis, and if the mouse is dragged horizontally, the molecule is rotated about the z-axis. The `direct` transform is then applied to every atom in the methane molecule.

Listing 17.4 The `MethaneApp` class instantiates a `Methane` object and rotates this object using mouse actions.

```
package org.opensourcephysics.sip.ch17;
import java.awt.event.MouseEvent;
import javax.swing.JFrame;
import org.opensourcephysics.frames.*;
import org.opensourcephysics.display.*;

public class MethaneApp implements InteractiveMouseHandler {
   DisplayFrame frame = new DisplayFrame("Methane");
   Methane methane = new Methane();
   double mouseX = 0, mouseY = 0;

   public MethaneApp() {
      frame.addDrawable(methane);
      frame.setPreferredMinMax(-1, 1, -1, 1);
      frame.setInteractiveMouseHandler(this);
      frame.setVisible(true);
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }

   public void handleMouseAction(InteractivePanel panel,
                                 MouseEvent evt) {
      switch(panel.getMouseAction()) {
      case InteractivePanel.MOUSE_DRAGGED:
         double dx = panel.getMouseX()-mouseX;
         double dy = panel.getMouseY()-mouseY;
         Rotation3D rotation = new Rotation3D(Math.sqrt(dx*dx+dy*dy),
              new double[] {dy, 0, dx});
         methane.transform(rotation);
         mouseX += dx;
         mouseY += dy;
         panel.repaint();
         break;
      case InteractivePanel.MOUSE_PRESSED:
```