Computational Physics and the Open Source Physics Project

Division of Computational Physics
San Diego, August 26, 2002

Harvey Gould
Clark University
hgould@clarku.edu
http://stp.clarku.edu/
http://www.opensourcephysics.org/

collaborators:
Wolfgang Christian, Davidson College
Jan Tobochnik, Kalamazoo College
Joshua Gould, University of Pittsburgh
Peter Sibley, Clark University

support: National Science Foundation

Goals and Nature of Project

1. The advantages of object oriented programming are well known, but it can be very time consuming to write all the necessary classes associated with graphical input and output.
2. Develop an extensive set of APIs that will make writing simulations in Java much easier, especially in the context of education.
3. Project is based on the GNU open source model.
4. Make available a large number of Java simulations in physics and encourage physicists and others to adopt a common codebase.

Integrating computational physics into the curriculum

1. Canned software such as *Interactive Physics* or programs for specific purposes from Physics Academic Software, <http://www.webassign.net/pasnew/>.
2. Give students sample problems and code using symbolic languages such as Mathematica, Maple, Matlab, or MathCad.
3. Use VPython and have students write computer simulations using powerful 3D graphics. VPython is free and open source. <http://vpython.org>

   Graphics statements are largely hidden from users. *Matter & Interactions*, Ruth Chabay and Bruce Sherwood, Wiley (2002).

4. *Physlets*, developed by Wolfgang Christian. Java applets built into Web pages using Javascript. Common user interface, ability of instructors to tailor physlets, and good sets of questions.

   Students do not usually learn what is behind the simulations, but physlets help student learning and introduce students to possibilities of simulations.

5. Java with Open Source Physics Library.

   A simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high performance, multithreaded, and dynamic language.

Nature of Open Source library

- Based on Swing and Java 1.3x. Recently, Java3D has been incorporated.

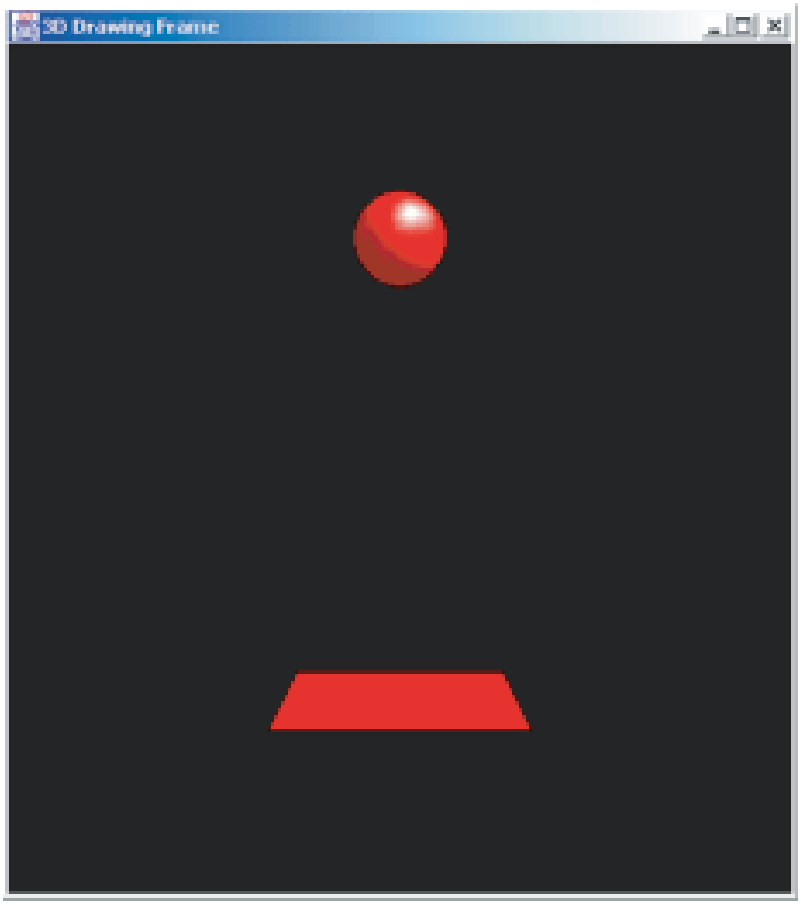
- Programs are written as applications, but can be incorporated into Web pages without any modifications.

```
<applet code="org.opensourcephysics.stp.applets.ApplicationApplet"
codebase ="../classes" width="150" height="40" align = "middle">
<param name="app" value ="org.opensourcephysics.sip.demon.DemonApp">
<param name="control"
      value ="org.opensourcephysics.controls.CalculationControl">
<param name="buttons" value ="setLogHistogram, log-linear plot">
</applet>
```

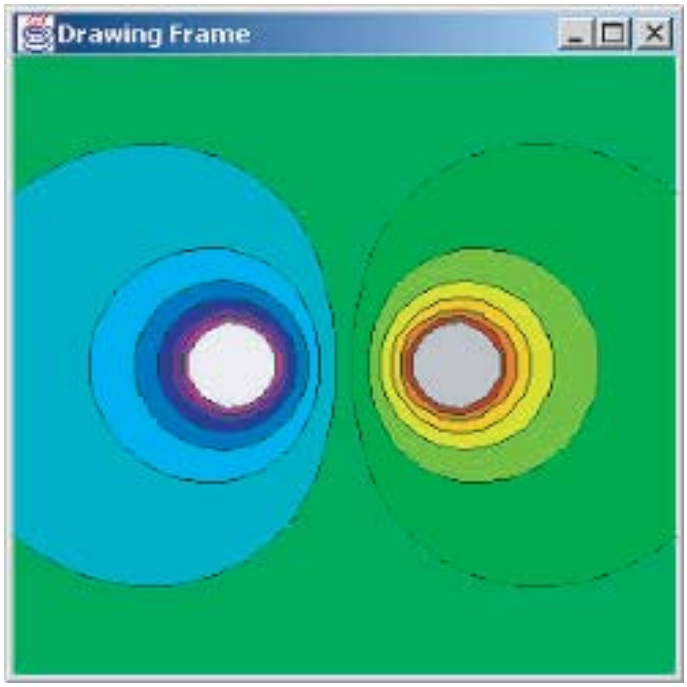
- Easy to do animations and plots.

```
public class CircleAndArrow {
   public static void main(String[] args) {
      // create a drawing frame and a drawing panel
      DrawingPanel panel = new DrawingPanel();
      DrawingFrame frame = new DrawingFrame(panel);
      // create a circle and an arrow
      Drawable circle = new Circle(0, 0);
      panel.addDrawable(circle);
      Arrow arrow = new Arrow(0, 0,4,3);
      panel.addDrawable(arrow);
      frame.show();
   }
```

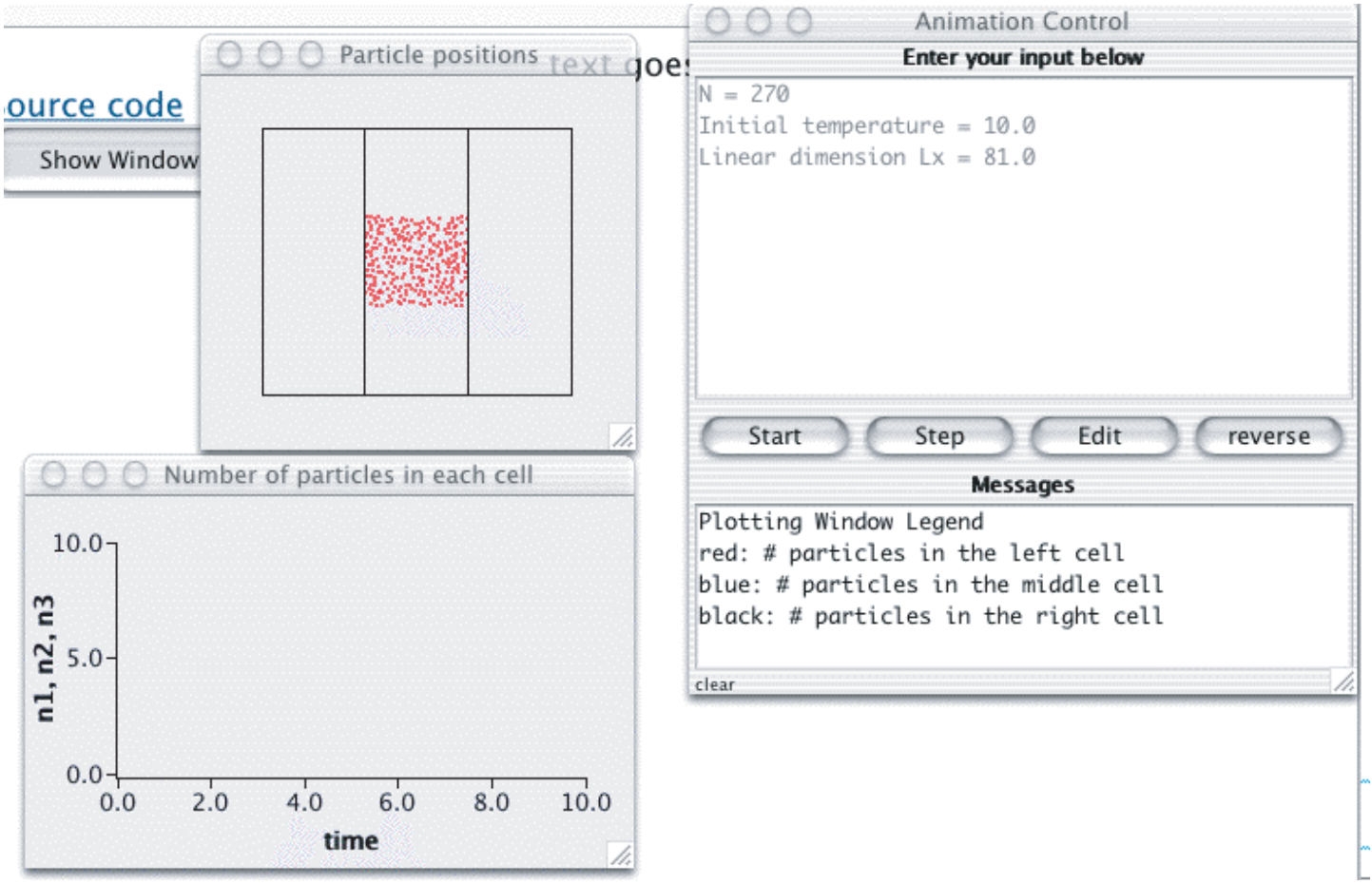
```
public class Ball {
   public static void main(String[] args) {
      DrawingPanel3D panel = new DrawingPanel3D();
      DrawingFrame3D frame = new DrawingFrame3D(panel);
      // Create the ball and floor
      DShape ball = new DSphere(-5.0f,0.0f,0.0f,0.5f,Color.yellow);
      panel.addDrawable3D(ball);
      DShape floor = new DBox(6.0f,0.0f,0.0f,0.2f,4.0f,4.0f,Color.green);
      panel.addDrawable3D(floor);
      // Shift the view back and show the frame
      panel.shiftSceneXYZ(0.0f,0.0f,-20.0f);
      frame.show();
   }
}
```



```
public class ContourApp {
   public static void main(String[] args) {
      // create the frame, the panel, and the field
      DrawingPanel drawingPanel = new DrawingPanel();
      DrawingFrame frame= new DrawingFrame(drawingPanel);
      Contour contour = new Contour();
      drawingPanel.addDrawable(contour);
      // set defaults
      contour.setZMinMax(-2,2);
      contour.setColorMode(Contour.SPECTRUM);
      // create some test data and add it to the contour
      double[][][] data = TestData.dipoleScalarField(32,32,-3,3,-3,3);
      contour.setDataArray(data);
      drawingPanel.repaint();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
}
```



Look and Feel



http://stp.clarku.edu/simulations/

Introduction to Computer Simulation Methods

In collaboration with Wolfgang Christian of Davidson College, Jan Tobochnik, and I are writing the third edition of our computer simulation text in **Java**.

http://sip.clarku.edu