```
/*
 * File: ShowData.java
 * Author: Nancy McCracken
 * Converted to Java 1.1: October 30, 1997
 * Copyright: Northeast Parallel Architectures Center
 *            at Syracuse University 1997
 *
 * This Java applet divides the applet area into two parts:  an area
 * with controls such as buttons and checkboxes, called controlPanelNorth,
 * and an area called dataPanel which is treated as a Canvas to display
 * a set of data points.  The dataPanel is a separate class.
 *
 * The data points are read from a file which has the following format:
 *    an integer which is the number of data points
 *    a maximum x value - the horizontal display will cover 0 through this value
 *    a maximum y value - the vertical display will cover 0 through this value
 *    data points x y
 *
 * This applet is intended to be a template for displaying a set of static
 * data from a file.  Each point is displayed in a very simple fashion, and
 * the only two controls provided is a choice box with 5 colors and a checkbox
 * group of 3 checkboxes controlling the shape of the point icon.
 *
 */

import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;

public class ShowData extends java.applet.Applet
                        implements ItemListener
{
  DataPanel dataPanel;          // the display panel, from the class below
  String inname = null;         // name of the data file

  Choice colorChoice;           // color control
  CheckboxGroup shape;          // shape controls
  Checkbox circleBox, squareBox, plusBox;

  public void init()
  { InputStream instream = null;
    Panel controlPanelNorth = new Panel();

    setLayout(new BorderLayout());

    // initialize the input stream and data panel
    try
      { inname = getParameter("inputfile");
        instream = new URL(getDocumentBase(), inname).openStream();
        dataPanel = new DataPanel(instream);
       }
    catch (IOException e)
      { System.err.println
          ("I/O Exception:  probably bad filename or bad connection."
                 + e.getMessage());
      }

  // make the control panel
  controlPanelNorth.setLayout(new FlowLayout(FlowLayout.LEFT, 5, 5));
```

```
  colorChoice = new Choice();
  colorChoice.addItem("green");
  colorChoice.addItem("cyan");
  colorChoice.addItem("blue");
  colorChoice.addItem("purple");
  colorChoice.addItem("magenta");
  colorChoice.addItemListener(this);

  shape = new CheckboxGroup();
  circleBox = new Checkbox("circle", shape, true);
  squareBox = new Checkbox("square", shape, false);
  plusBox = new Checkbox("plus", shape, false);
  circleBox.addItemListener(this);
  squareBox.addItemListener(this);
  plusBox.addItemListener(this);

  controlPanelNorth.add(colorChoice);
  controlPanelNorth.add(circleBox);
  controlPanelNorth.add(squareBox);
  controlPanelNorth.add(plusBox);

  // add the two components to the applet area
  add("North", controlPanelNorth);
  add("Center", dataPanel);
  }

  public void update(Graphics g)
  {
    dataPanel.repaint();
  }

  public void paint(Graphics g)
  {
    dataPanel.repaint();
  }

  public void itemStateChanged(ItemEvent e)
  {
  /* in response to activation of controls, call methods in the
        DataPanel class that will change the appearance of the display */

    if (e.getSource() == colorChoice)
      {
        dataPanel.setColor(colorChoice.getSelectedIndex());
        repaint();
      }
    else // it was a checkbox
      {
        dataPanel.setShape(shape.getSelectedCheckbox().getLabel());
        repaint();
      }
  }
}

class DataPanel extends Panel
{
  // variables to hold the data from the file
  int numdata;
  double maxx, maxy;
  double xdata[], ydata[];
```

```
  // variables to scale the data to pixels
  int maxxoffset, maxyoffset;
  int xoffset[], yoffset[];
  int xscale,yscale;

  InputStreamReader isr;
  // these two variables control the current appearance of the data
  Color currentColor;
  String currentshape;

  // the set of allowed colors
  Color ctable[] = new Color[5];


  public DataPanel(InputStream instream)
  {
    try
    {
      isr = new InputStreamReader(instream);
      StreamTokenizer intokens = new StreamTokenizer(isr);
      intokens.parseNumbers();  intokens.slashSlashComments(true);

      /* read the number of data points and allocate data and pixel
           offset arrays */
      int tok = intokens.nextToken();
      numdata = (int) intokens.nval;
      xdata = new double[numdata]; ydata = new double[numdata];
      xoffset = new int[numdata]; yoffset = new int[numdata];

      /* read maximum x and y data values and scale data from 0 to maxx
        to pixels to fit the window, and same for y */
      tok = intokens.nextToken();
      maxx = intokens.nval;
      tok = intokens.nextToken();
      maxy = intokens.nval;
      maxxoffset = 600;
      maxyoffset = 450;
      xscale = (int) (maxxoffset/maxx);
      yscale = (int) (maxyoffset/maxy);

      // read x,y data pairs
      for (int i=0; i<numdata; i++)
      { tok = intokens.nextToken();  xdata[i] = intokens.nval;
        tok = intokens.nextToken();  ydata[i] = intokens.nval;
      }
/* for debugging
      for (int i=0; i<numdata; i++)
      { System.out.println(xdata[i]);
        System.out.println(ydata[i]);
      }
*/
    }
    catch (IOException e)
      { System.err.println
          ("I/O Exception:  Reading data from file"
                 + e.getMessage());
      }

    // initialize current control variables and color table
    currentColor = Color.green;
    currentshape = "circle";
```

```
    setBackground(Color.black);

    ctable[0] = Color.green;
    ctable[1] = Color.cyan;
    ctable[2] = Color.blue;
    ctable[3] = new Color(127,0,255);
    ctable[4] = Color.magenta;
  }


  public void setColor(int colorindex)
  { currentColor = ctable[colorindex];
  }


  public void setShape(String selectedshape)
  { currentshape = selectedshape;
  }


  public void paint(Graphics g)
  {
    g.setColor (currentColor);
    if (currentshape.equals("circle"))
      { // draw a circle for each point
        for (int i=0; i<numdata; i++)
        { g.fillOval((int)(xdata[i]*xscale),
                maxyoffset - (int)(ydata[i]*yscale), 12, 12);
        }
      }
    else if (currentshape.equals("square"))
      { // draw a square for each point
        for (int i=0; i<numdata; i++)
        { g.fillRect((int)(xdata[i]*xscale),
                maxyoffset - (int)(ydata[i]*yscale), 12, 12);
        }
      }
    else if (currentshape.equals("plus"))
      { //draw a plus for each point
        g.setFont(new Font("TimesRoman", Font.BOLD, 14));
        for (int i=0; i<numdata; i++)
        { g.drawString("+", (int)(xdata[i]*xscale),
                maxyoffset - (int)(ydata[i]*yscale));
        }
      }
  }
}
```