```
/*-------------------------------------------------------------
 Displaying live YTD stock prices using Yahoo as data source,
 and mimicing a graph style akin to what might be shown at
 http://finance.yahoo.com/echarts?s=GOOG

 Author: Michael Goldwasser
 -------------------------------------------------------------*/

String symbol = "YHOO";   // initial choice
String stockName, caption; // set dynamically based on symbol

int padding = 50;  // margin between canvas edge and graph rectangle on each edge
int X1, Y1, X2, Y2;    // coordinate range for the graph

Table data;
float low, high, incr;  // relevant range of stock prices

void setup() {
  // drawing setup
  size(800, 600);

  X1 = padding;
  Y1 = padding;
  X2 = width - padding;
  Y2 = height - padding;
  textFont(createFont("SansSerif", padding/2));
  loadData();
}

void loadData() {
  symbol = symbol.toUpperCase();

  // Load the full name of the stock for caption
  String url = "http://finance.yahoo.com/d/quotes.csv?f=n&s=";
  url += symbol;
  String full = loadStrings(url)[0];

  if (full.equals("N/A")) {
    caption = "unknown (" + symbol + ")";
  } else {
    stockName = full.substring(1, full.length()-1);  // remove quotation marks
    caption = stockName + " (" + symbol + ")";

    // Load the historical data
    url = "http://ichart.yahoo.com/table.csv?s=";
    url += symbol;
    url += "&a=0&b=1&c=1900";  // start date
    url += "&d=" + (month()-1);
    url += "&e=" + day();
    url += "&f=" + year();
    url += "&ignore=.csv";
    data = loadTable(url, "header");

    low = high = data.getFloat(0, "Adj Close");
    for (int j=1; j < data.getRowCount(); j++) {
      float price = data.getFloat(j, "Adj Close");
      low = min(low, price);
      high = max(high, price);
    }

    computeIncrements();
  }

  symbol = "";  // reset for next time
}

```

```
 67:
 68: // Determine vertical increment levels and rounded high/low values for graphing
 69: void computeIncrements() {
 70:   float gap = (high - low)/8;   // ideally we want 8 sections.
 71:   int digits = nf(int(gap)).length();
 72:   int power = int(pow(10,digits-1));
 73:   incr = int(0.5 + gap/power)*power;
 74:   low = int(low/incr)*incr;
 75:   high = int(1 + high/incr)*incr;
 76: }
 77:
 78: void showGraph() {
 79:   background(255);
 80:   strokeWeight(1);
 81:   stroke(0, 150, 233);
 82:   fill(240, 250, 255);
 83:   beginShape();
 84:   vertex(X1, Y2);
 85:   vertex(X2, Y2);
 86:   int n = data.getRowCount();  // number of days of trading
 87:   for (int j=0; j < n; j++) {
 88:     float price = data.getFloat(j, "Adj Close");
 89:     vertex(map(j, 0, n-1, X2, X1), map(price, low, high, Y2, Y1));
 90:   }
 91:   endShape();
 92:
 93:   // draw guide lines and labels
 94:   textAlign(LEFT, BOTTOM);
 95:   stroke(127);
 96:   fill(127);
 97:   textSize(10);
 98:   for (float price = low; price < high; price += incr) {
 99:     float y = map(price,low,high,Y2,Y1);
100:     line(X1,y,width,y);
101:     text(nf(price,1,2), X2 + 10, y);
102:   }
103:
104:   // draw bounding rectangle
105:   stroke(0);
106:   noFill();
107:   rect(X1, Y1, X2-X1, Y2-Y1);
108: }
109:
110:
111: // Display interactive crosshairs for day indicated with mouse
112: void showInteractive() {
113:   // determine y-value for relevant stock entry
114:   int j = int(map(mouseX, X1, X2, data.getRowCount()-1, 0));
115:   float price = data.getFloat(j, "Adj Close");
116:   float y = map(price, low, high, Y2, Y1);
117:
118:   stroke(0);
119:   // dashed lines
120:   strokeWeight(1);
121:   for (int a=X1; a < X2; a += 5) {
122:     line(a, y, a+3, y);
123:   }
124:   for (int b=Y1; b < Y2; b += 5) {
125:     line(mouseX, b, mouseX, b+3);
126:   }
127:
128:   // circle highlight
129:   strokeWeight(6);
130:   point(mouseX, y);
131:
132:   showCurrentPrice(y, price);
133:   showCurrentDate(mouseX, data.getString(j, "Date"));
134:   showCurrentData(j);
135: }
136:
137:
```

```
138: void showCurrentPrice(float y, float price) {
139:   textSize(10);
140:   String disp = nf(price, 1, 2);
141:   float w = 4+textWidth(disp);
142:   strokeWeight(0);
143:   stroke(50);
144:   fill(50);
145:   triangle(X2, y, X2+7, y+7, X2+7, y-7);
146:   rect(X2+7, y-7, w, 14);
147:
148:   fill(255);
149:   textAlign(LEFT, CENTER);
150:   text(disp, X2+9, y);
151: }
152:
153: void showCurrentDate(float x, String date) {
154:   textSize(10);
155:   float w = 4+textWidth(date);
156:   strokeWeight(0);
157:   stroke(50);
158:   fill(50);
159:   rect(x-w/2, Y2, w, 14);
160:
161:   fill(255);
162:   textAlign(CENTER, TOP);
163:   text(date, x, Y2+2);
164: }
165:
166:
167: String fields[] = {"Open", "Close", "Low", "High", "Volume"};
168: String fieldsDisplay[] = {"Open", "Close", "Low", "High", "Vol", "% Chg"};
169:
170: // Show stock data for day with index j
171: void showCurrentData(int j) {
172:   float minipad = 5;
173:   fill(255);
174:   stroke(0);
175:   strokeWeight(1);
176:   textSize(10);
177:   float w = textWidth("% chg  XXXXX.XX%");
178:   rect(X1+minipad, Y1+minipad, w + 2*minipad, 72 + 2*minipad);
179:   float ctrX = X1 + 2*minipad + textWidth("% chg ");
180:   fill(127);
181:   textAlign(RIGHT, TOP);
182:   for (int a=0; a < fieldsDisplay.length; a++) {
183:     text(fieldsDisplay[a]+" ", ctrX, Y1 + 2*minipad + 12*a);
184:   }
185:
186:   textAlign(LEFT, TOP);
187:   float adj = data.getFloat(j, "Adj Close") / data.getFloat(j, "Close");
188:   for (int a=0; a < fields.length; a++) {
189:     float val = data.getFloat(j, fields[a]);
190:     String valDisplay;
191:     if (a < fields.length-1) {
192:       valDisplay = nf(adj*val, 1, 2);
193:     } else {
194:       valDisplay = nf(val/1000000, 1, 2) + "M";  // volume in millions
195:     }
196:     text(" " + valDisplay, ctrX, Y1 + 2*minipad + 12*a);
197:   }
198:   float pctChange = 100 * data.getFloat(j, "Adj Close") /
199:                     data.getFloat(data.getRowCount()-1, "Adj Close") - 100;
200:   text(" " + nf(pctChange, 1, 2) + "%", ctrX, Y1 + 2*minipad + 60);
201: }
202:
```

```
203:
204: void draw() {
205:   if (data != null) {
206:     showGraph();
207:     if (mouseX >= X1 && mouseX <= X2 && mouseY >= Y1 && mouseY <= Y2) {
208:       showInteractive();
209:     }
210:   }
211:
212:   textAlign(LEFT, CENTER);
213:   textSize(padding/2);
214:   if (symbol.length() == 0) {
215:     fill(0);
216:     text(caption, X1, padding/2);
217:   } else {
218:     fill(255, 0, 0);
219:     text(symbol.toUpperCase(), X1, padding/2);
220:   }
221: }
222:
223:
224: void keyPressed() {
225:   if (key == ENTER || key == RETURN) {
226:     loadData(); // load data for current symbol
227:   } else if ((key >= 'a' && key <= 'z') || (key >= 'A' && key <= 'Z')) {
228:     symbol += key;
229:   }
230: }
```