```
// StopWatch -- pull together a bunch of different Java concepts.

import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;

public class StopWatch extends Applet {
  private TickerThread ticker;
  private Button go, stop;
  private ClockFace face;
  private boolean going;

  public void init() {
    face = new ClockFace();
    face.setBackground( Color.blue );

    ticker = new TickerThread(face);

    go = new Button( "Go" );
    go.addActionListener( new TickStarter(ticker, this) );

    stop = new Button( "Stop" );
    stop.addActionListener( new TickStopper(ticker, this) );

    Panel p = new Panel();
    p.setLayout( new GridLayout( 2, 1 ) );
    p.add( go );
    p.add( stop );

    setLayout(new BorderLayout() );
    add( p, BorderLayout.WEST );
    add( face, BorderLayout.CENTER );

    going = false;
  }

  public void start() {
    if (going)
      ticker.resume();
  }

  public void stop() {
    if (going)
      ticker.suspend();
  }

  public void setGoing(boolean g) {
    going = g;
```

```
  }
}

class ClockFace extends Canvas {
  private int seconds, minutes;
  private final int CENTER_X = 100;
  private final int CENTER_Y = 100;
  private final int RADIUS = 90;

  public ClockFace() {
    seconds = 0;
    minutes = 0;
  }

  public void paint( Graphics g ) {
    g.drawOval( CENTER_X - RADIUS, CENTER_Y - RADIUS,
                2 * RADIUS, 2 * RADIUS);
    drawHand( g, (int) (0.95 * RADIUS), seconds ); // second hand
    drawHand( g, (int) (0.75 * RADIUS), minutes ); // second hand
  }

  public void tick() {
    seconds++;
    if (seconds == 60) {
      minutes++;
      seconds = 0;
    }
    if (minutes == 60) {
      minutes = 0;
    }
    repaint();
  }

  private void drawHand( Graphics g, int length, int time) {
    double angle = (double) time / 60.0 * 2 * Math.PI;
    g.drawLine(CENTER_X, CENTER_Y,
               (int) ( CENTER_X + (length * Math.sin(angle)) ),
               (int) ( CENTER_X - (length * Math.cos(angle)) ) );
  }
}


class TickerThread extends Thread {
  private ClockFace face;

  public TickerThread(ClockFace f) {
    face = f;
  }
```

```
  public void run() {
    while (true) {
      try {
        sleep(1000);
      }
      catch (InterruptedException e) {
        System.err.println("Sleep interrupted");
      }
      face.tick();
    }
  }
}

class TickStarter implements ActionListener {
  private TickerThread ticker;
  private StopWatch watch;

  public TickStarter(TickerThread t, StopWatch w) {
    ticker = t;
    watch = w;
  }

  public void actionPerformed( ActionEvent aevt ) {
    watch.setGoing(true);
    if (!ticker.isAlive()) {
      ticker.start();
    } else {
      ticker.resume();
    }
  }
}

class TickStopper implements ActionListener {
  private TickerThread ticker;
  private StopWatch watch;

  public TickStopper(TickerThread t, StopWatch w) {
    ticker = t;
    watch = w;
  }

  public void actionPerformed( ActionEvent aevt ) {
    watch.setGoing(false);
    if (ticker.isAlive()) {
      ticker.suspend();
    }
  }
```

```
}
```