```
            if(d<=0) {
               d += roadLength;
            }
            if(v[i]>=d) { // slow down due to cars in front
               v[i] = d-1;
            }
            if((v[i]>0)&&(Math.random()<p)) { // randomization
               v[i]--;
            }
            x[i] = (xtemp[i]+v[i])%roadLength;
            flow += v[i];
         }
         steps++;
         computeSpaceTimeDiagram();
      }

      public void computeSpaceTimeDiagram() {
         t++;
         if(t<scrollTime) {
            for(int i = 0;i<numberOfCars;i++) {
               spaceTime.setValue(x[i], t, 1);
            }
         } else { // scroll diagram
            for(int y = 0;y<scrollTime-1;y++) {
               for(int i = 0;i<roadLength;i++) {
                  spaceTime.setValue(i, y, spaceTime.getValue(i, y+1));
               }
            }
            for(int i = 0;i<roadLength;i++) {
               // remove cars from road
               spaceTime.setValue(i, scrollTime-1, 0);
            }
            for(int i = 0;i<numberOfCars;i++) {
               spaceTime.setValue(x[i], scrollTime-1, 1); // add cars
            }
         }
      }

      public void draw(DrawingPanel panel, Graphics g) {
         if(x==null) {
            return;
         }
         road.setBlock(0, 0, new byte[roadLength][1]);
         for(int i = 0;i<numberOfCars;i++) {
            road.setValue(x[i], 0, (byte) 1);
         }
         road.draw(panel, g);
         g.drawString("Number of Steps = "+steps, 10, 20);
         g.drawString("Flow = "+
               ControlUtils.f3((double)flow/(roadLength*steps)), 10, 40);
         g.drawString("Density = "+
               ControlUtils.f3((double)numberOfCars/(roadLength)), 10, 60);
      }
   }
```


The target class `FreewayApp` shows the movement of the cars and a space-time diagram, with time on the vertical axis and space on the horizontal axis. When the number of iterations equals `scrollTime`, the diagram scrolls down. The flow rate is the average of the car velocities divided by the length of the highway. Thus, two cars moving at constant velocity will have twice the flow rate of one car moving at the same velocity.

Listing 14.4 `FreewayApp` Class.

```
package org.opensourcephysics.sip.ch14.traffic;
import org.opensourcephysics.controls.*;
import org.opensourcephysics.frames.*;

public class FreewayApp extends AbstractSimulation {
   Freeway freeway = new Freeway();
   DisplayFrame display = new DisplayFrame("Freeway");
   LatticeFrame spaceTime = new LatticeFrame("space", "time",
        "Space Time Diagram");

   public FreewayApp() {
      display.addDrawable(freeway);
   }

   public void initialize() {
      freeway.numberOfCars = control.getInt("Number of cars");
      freeway.roadLength = control.getInt("Road length");
      freeway.p = control.getDouble("Slow down probability");
      freeway.maximumVelocity = control.getInt("Maximum velocity");
      display.setPreferredMinMax(0, freeway.roadLength, -3, 4);
      freeway.initialize(spaceTime);
   }

   public void doStep() {
      freeway.step();
   }

   public void reset() {
      control.setValue("Number of cars", 10);
      control.setValue("Road length", 50);
      control.setValue("Slow down probability", 0.5);
      control.setValue("Maximum velocity", 2);
      control.setValue("Steps between plots", 1);
      enableStepsPerDisplay(true);
   }

   public void resetAverages() {
      freeway.flow = 0;
      freeway.steps = 0;
   }

   public static void main(String[] args) {
      SimulationControl control =
           SimulationControl.createApp(new FreewayApp());
      control.addButton("resetAverages", "resetAverages");
   }
}
```