package chapter3;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.Font;

import java.awt.Graphics;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.Timer;

/\*\* TrafficAnimation.java

\* CS 121 Project 1: Traffic Animation

\*/

/\*\*

\* Animates a [put your description here]

\* @author [put your name here]

\*/

@SuppressWarnings("serial")

public class TrafficAnimation extends JPanel {

//Note: This is where you declare constants and variables that

// need to keep their values between calls to paintComponent().

// Any other variables should be declared locally, in the

// method where they are used.

//constant to regulate the frequency of Timer events

// Note: 100ms is 10 frames per second - you should not need

// a faster refresh rate than this

private final int DELAY = 100; //milliseconds

//anchor coordinate for drawing / animating

private int x = 0;

//pixels added to x each time paintComponent() is called

private int x2= 600;

private int stepSize = 10;

/\* This method draws on the applet's Graphics context.

\* This is where the majority of your work will be.

\*

\* (non-Javadoc)

\* @see java.awt.Container#paint(java.awt.Graphics)

\*/

public void paintComponent(Graphics canvas)

{

//clears the previous image

//super.paintComponent(canvas);

//account for changes to window size

int width = getWidth(); // panel width

int height = getHeight(); // panel height

String str = "Can I go yet!";

canvas.setFont(new Font("Serif",Font.BOLD,width/16));

//setBackground(Color.yellow);

//Fill the canvas with the background color

canvas.setColor(getBackground());

canvas.fillRect(0, 0, width, height);

//Calculate the new position

x = (x + stepSize) % width;

//Draw new square

//TODO: replace this square with your drawing

int squareSide = height/5;

int y = height/2 - squareSide/2;

canvas.setColor(Color.darkGray);

canvas.fillRect(0, height/2-squareSide, width, squareSide\*2);

canvas.setColor(Color.green);

canvas.fillRect(0,height/2+squareSide , width, squareSide\*2);

if(x2<=0) x2=width;

else{

x2=(x2-stepSize) % width;

}

canvas.setColor(Color.white);

canvas.fillRect(x2, height/2, 20, 10);

canvas.fillRect(x2+30,height/2,20,10);

canvas.fillRect(x2+60,height/2, 20,10);

canvas.fillRect(x2+90,height/2, 20,10);

canvas.fillRect(x2+120,height/2, 20,10);

canvas.fillRect(x2+150,height/2, 20,10);

canvas.fillRect(x2+180,height/2, 20,10);

canvas.fillRect(x2+210,height/2, 20,10);

canvas.fillRect(x2+240,height/2, 20,10);

canvas.fillRect(x2+270,height/2, 20,10);

canvas.fillRect(x2+310,height/2, 20,10);

canvas.fillRect(x2+340,height/2, 20,10);

canvas.fillRect(x2+370,height/2, 20,10);

canvas.fillRect(x2+410,height/2, 20,10);

canvas.fillRect(x2+440,height/2, 20,10);

canvas.fillRect(x2+470,height/2, 20,10);

canvas.fillRect(x2+510,height/2, 20,10);

canvas.fillRect(x2-30,height/2,20,10);

canvas.fillRect(x2-60,height/2, 20,10);

canvas.fillRect(x2-90,height/2, 20,10);

canvas.fillRect(x2-120,height/2, 20,10);

canvas.fillRect(x2-150,height/2, 20,10);

canvas.fillRect(x2-180,height/2, 20,10);

canvas.fillRect(x2-210,height/2, 20,10);

canvas.fillRect(x2-240,height/2, 20,10);

canvas.fillRect(x2-270,height/2, 20,10);

canvas.fillRect(x2-310,height/2, 20,10);

canvas.fillRect(x2-340,height/2, 20,10);

canvas.fillRect(x2-370,height/2, 20,10);

canvas.fillRect(x2-410,height/2, 20,10);

canvas.fillRect(x2-440,height/2, 20,10);

canvas.fillRect(x2-470,height/2, 20,10);

canvas.fillRect(x2-510,height/2, 20,10);

canvas.setColor(Color.blue);

canvas.fillRect(x2,y-(squareSide/2), squareSide, squareSide/2);//blue cart

canvas.setColor(Color.gray);

canvas.fillOval(x2, y, squareSide/3, squareSide/3);// wheel for blue cart

canvas.fillOval(x2+squareSide/2,y,squareSide/3, squareSide/3);

canvas.setColor(Color.green);

canvas.fillRect(x, y+(squareSide/2), squareSide, squareSide/2);//green cart

canvas.setColor(Color.GRAY);

canvas.fillOval(x, y+squareSide, squareSide/3, squareSide/3);//grey wheel

canvas.setColor(Color.GRAY);

canvas.fillOval(x+squareSide/2, y+squareSide, squareSide/3, squareSide/3);

canvas.setColor(Color.white);

canvas.drawArc(x,y+squareSide ,squareSide/3 , squareSide/3, x2, 60);

canvas.drawArc(x,y+squareSide ,squareSide/3 , squareSide/3, x2+120, 60);

canvas.drawArc(x,y+squareSide ,squareSide/3 , squareSide/3, x2+240, 60);

Color green =new Color(1,188,4);

canvas.setColor(green);

canvas.drawString(str, width/2, height/6);

canvas.drawLine(width/2, (height/4)\*3, width/2,(height/4)\*3+100 );// avatar

canvas.drawOval((width/2)-20,(height/4)\*3 , 40, 40);

canvas.setColor(Color.white);

canvas.fillOval(width/2-40, (height/4)\*3+40, 80, 60);

canvas.fillOval(width/6, height/8, 80, 40);

canvas.fillOval(width/15, height/9, 80, 40);

canvas.setColor(Color.red);

canvas.fillOval(-40, -40, 80, 80);

}

/\*\*

\* Constructor for the display panel initializes

\* necessary variables. Only called once, when the

\* program first begins.

\* This method also sets up a Timer that will call

\* paint() with frequency specified by the DELAY

\* constant.

\*/

public TrafficAnimation()

{

setBackground(Color.blue);

//Do not initialize larger than 800x600

int initWidth = 400;

int initHeight = 400;

setPreferredSize(new Dimension(initWidth, initHeight));

this.setDoubleBuffered(true);

//Start the animation - DO NOT REMOVE

startAnimation();

}

/////////////////////////////////////////////

// DO NOT MODIFY main() or startAnimation()

/////////////////////////////////////////////

/\*\*

\* Starting point for the TrafficAnimation program

\* @param args unused

\*/

public static void main (String[] args)

{

JFrame frame = new JFrame ("Traffic Animation");

frame.setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().add(new TrafficAnimation());

frame.pack();

frame.setVisible(true);

}

/\*\*

\* Create an animation thread that runs periodically

\* DO NOT MODIFY this method!

\*/

private void startAnimation()

{

ActionListener taskPerformer = new ActionListener() {

public void actionPerformed(ActionEvent event) {

repaint();

}

};

new Timer(DELAY, taskPerformer).start();

}

}