**12주차**

20234014 컴퓨터 공학과 이은정

1.

package week12\_01;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Graphics;

import java.awt.GridLayout;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

public class SnowManFace extends JFrame {

    JLabel text;

    int stress = 0;

    class MyPanel extends JPanel {

        public void paintComponent(Graphics g) {

            super.paintComponent(g);

            if (0 >= stress) {

                g.setColor(Color.YELLOW);

                g.fillOval(20, 30, 200, 200);

                g.setColor(Color.BLACK);

                g.drawArc(60, 80, 50, 50, 180, -180);

                // 왼쪽 눈

                g.drawArc(150, 80, 50, 50, 180, -180);

                // 오른쪽 눈

                g.drawArc(70, 130, 100, 70, 180, 180);

                // 입

                g.setColor(Color.green);

            } else {

                g.setColor(Color.RED);

                g.fillOval(20, 30, 200, 200);

                g.setColor(Color.BLACK);

                g.drawArc(60, 80, 50, 50, 180, 180);

                // 왼쪽 눈

                g.drawArc(150, 80, 50, 50, 180, 180);

                // 오른쪽 눈

                g.drawArc(70, 130, 100, 70, 180, -180);

                // 입

                g.setColor(Color.green);

            }

        }

    }

    public SnowManFace() {

        setSize(300, 300);

        setTitle("눈사람 얼굴");

        this.setLayout(new BorderLayout());

        setVisible(true);

        JPanel stressText = new JPanel();

        text = new JLabel("현재 스트레스 지수=" + stress);

        stressText.add(text);

        this.add(stressText, BorderLayout.NORTH);

        JPanel buttons = new JPanel();

        buttons.setLayout(new GridLayout(1, 3));

        JButton present = new JButton("선물");

        present.addActionListener(e -> {

            stress -= 1;

            text.setText("현재 스트레스 지수=" + stress);

            repaint();

        });

        present.setBackground(Color.green);

        JButton assignments = new JButton("과제");

        assignments.addActionListener(e -> {

            stress += 1;

            text.setText("현재 스트레스 지수=" + stress);

            repaint();

        });

        assignments.setBackground(Color.red);

        buttons.add(present);

        buttons.add(assignments);

        this.add(buttons, BorderLayout.SOUTH);

        add(new MyPanel(), BorderLayout.CENTER);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

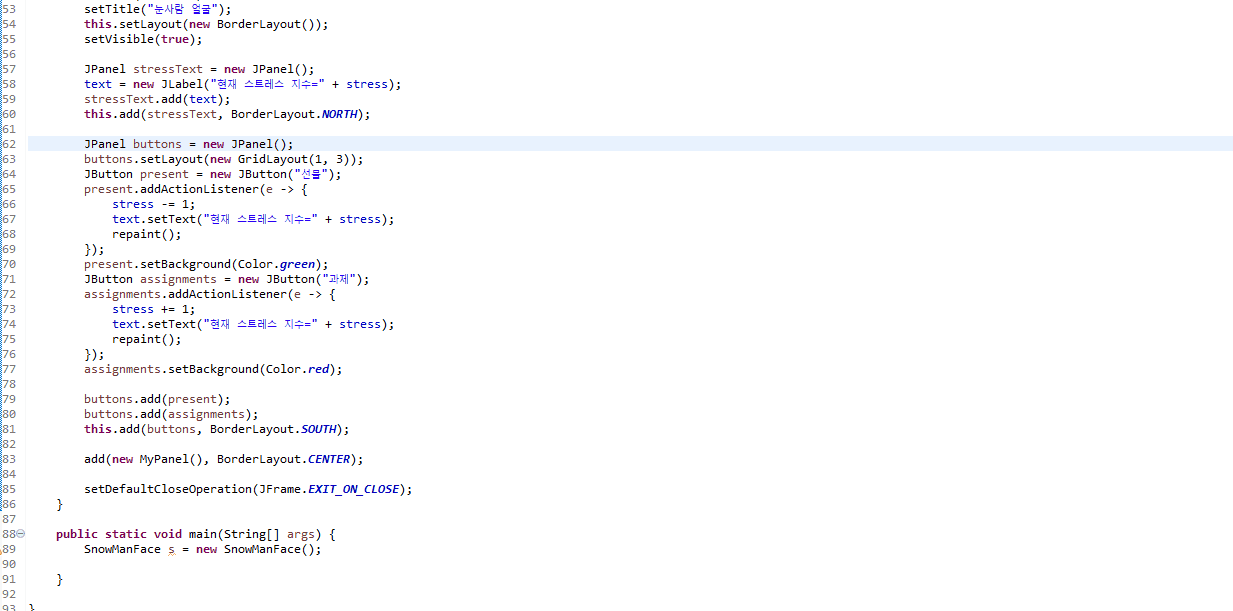
    }

    public static void main(String[] args) {

        SnowManFace s = new SnowManFace();

    }

}



해당 프로그램은 각 판넬에 각자의 요소를 보여주는 프로그램이다. JFrame에는 3개의 판넬이 있고, 첫 번째 판넬에서는 스트레스 지수를, 가운데 판넬은 얼굴을, 마지막 판넬은 버튼을 보여주는 프로그램이다. Int stress에 기본 값인 0을 넣어두고, 각 버튼에서는 stress+=1 or stress-=1 를 통해 스트레스 값을 조정하고, repaint를 한다. 얼굴은 if stress>0일 경우, 얼굴을 빨갛게, 눈을 그리는 방식을 다르게 하여 그리고, 그 외에는 기존 방식을 따르게 한다.

2. 신호등

package week12\_02;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Graphics;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JPanel;

class MyFrame extends JFrame {

    int n = 0;

    int r = 150;

    MyFrame() {

        this.setTitle("신호등");

        this.setSize(600, 300);

        this.add(new MyPanel());

        this.setVisible(true);

        this.setDefaultCloseOperation(this.EXIT\_ON\_CLOSE);

    }

    class MyPanel extends JPanel implements ActionListener {

        int light\_number = 0;

        public MyPanel() {

            setLayout(new BorderLayout());

            JButton b = new JButton("신호 변경");

            b.addActionListener(e -> {

                n += 1;

                if (n == 3)

                    n = 0;

                repaint();

            });

            b.addActionListener(this);

            add(b, BorderLayout.SOUTH);

        }

        @Override

        protected void paintComponent(Graphics g) {

            super.paintComponent(g);

            int x = 50;

            if (n == 0) {

                g.setColor(Color.RED);

                g.fillOval(x, r / 3, r, r);

                g.setColor(Color.BLACK);

                x += r;

                g.drawOval(x, r / 3, r, r);

                x += r;

                g.drawOval(x, r / 3, r, r);

            } else if (n == 1) {

                g.setColor(Color.BLACK);

                g.drawOval(x, r / 3, r, r);

                x += r;

                g.setColor(Color.YELLOW);

                g.fillOval(x, r / 3, r, r);

                x += r;

                g.setColor(Color.BLACK);

                g.drawOval(x, r / 3, r, r);

            } else if (n == 2) {

                g.setColor(Color.BLACK);

                g.drawOval(x, r / 3, r, r);

                x += r;

                g.drawOval(x, r / 3, r, r);

                x += r;

                g.setColor(Color.GREEN);

                g.fillOval(x, r / 3, r, r);

            }

        }

        @Override

        public void actionPerformed(ActionEvent arg0) {

        }

    }

}

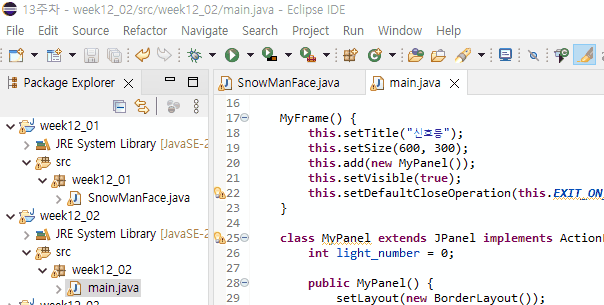
public class main {

    public static void main(String[] args) {

        MyFrame m = new MyFrame();

    }

}



해당 코드는 신호 변경을 통해 int n을 증가시키고, 해당 n에 따라 신호등을 출력하는 코드이다. JPanel에서 @paintComponent를 통해 각 요소를 그린다. 이 때, super(g);를 통해 기존 메서드를 초기화 하고, 그림을 그린다. 만약 그림을 그리다 n==3이 될 시, n을 0으로 초기화 한 이후, repaint()를 통해 다시 그림을 그린다.

3. 그림 그리기

package week12\_03;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Graphics;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.awt.event.MouseMotionAdapter;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JPanel;

class myFrame extends JFrame {

    int prevX, prevY, currX, currY;

    Boolean dragging;

    Color currentColor;

    myFrame() {

        this.setTitle("Simple Paint");

        this.setSize(500, 550);

        this.setLayout(new BorderLayout());

        JPanel canvas = new JPanel();

        canvas.setBackground(Color.white);

        canvas.addMouseListener(new MouseAdapter() {

            @Override

            public void mousePressed(MouseEvent e) {

                prevX = e.getX();

                prevY = e.getY();

                dragging = true;

            }

            @Override

            public void mouseReleased(MouseEvent e) {

                dragging = false;

            }

        });

        canvas.addMouseMotionListener(new MouseMotionAdapter() {

            @Override

            public void mouseDragged(MouseEvent e) {

                if (dragging) {

                    currX = e.getX();

                    currY = e.getY();

                    Graphics g = canvas.getGraphics();

                    g.setColor(currentColor);

                    g.drawLine(prevX, prevY, currX, currY);

                    prevX = currX;

                    prevY = currY;

                }

            }

        });

        this.add(canvas, BorderLayout.CENTER);

        JPanel buttons = new JPanel();

        JButton black = new JButton("Black");

        black.addActionListener(e -> {

            currentColor = Color.BLACK;

        });

        buttons.add(black);

        JButton red = new JButton("Red");

        red.addActionListener(e -> {

            currentColor = Color.RED;

        });

        buttons.add(red);

        JButton clear = new JButton("Clear");

        clear.addActionListener(e -> {

             repaint();

        });

        buttons.add(clear);

        this.add(buttons, BorderLayout.SOUTH);

        this.setLocationRelativeTo(null);

        this.setVisible(true);

        this.setDefaultCloseOperation(this.EXIT\_ON\_CLOSE);

    }

}

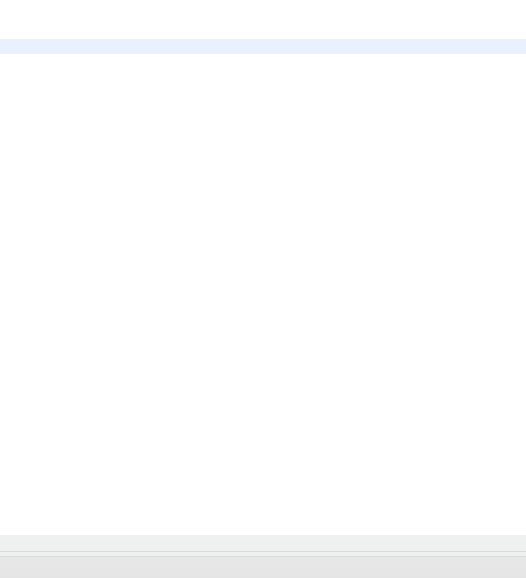
public class main {

    public static void main(String[] args) {

        myFrame m = new myFrame();

    }

}



위 메서드는 JPanel위에서 그림을 그리는 메서드이다. 그림을 그리는 메서드는 pdf에 나와있었기 때문에 버튼에 대해서만 설명하겠다. JFrame은 borderLayout의 SOUTH에 위치해 있다. 이 panel에는 balck, red, clear라는 버튼이 존재한다. Black, Red 버튼을 누르면 currentColor 이라는 Color 객체에 Color.BLACK, Color.RED 라는 각각의 색을 저장한다. Clear이라는 버튼을 누르면 repaint()를 통해 판넬을 초기화 한다.

4. 몬스터 캐치 게임

package week12\_04;

import java.awt.Graphics;

import java.awt.Image;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.util.ArrayList;

import java.util.Iterator;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.Timer;

class Monster {

    int x, y, hp;

    Image img;

    int speed;

    Monster(int x, int y, int hp) {

        this.x = x;

        this.y = y;

        this.hp = hp;

        ImageIcon icon = new ImageIcon(getClass().getResource("/Image/monster.png"));

        img = icon.getImage();

        speed = (int)(Math.random()\*10+5);

        if (Math.random()<0.5)

            speed\*=-1;

    }

    public void draw(Graphics g) {

        g.drawImage(img, x, y, null);

    }

}

class MyFrame extends JFrame {

    static ArrayList<Monster> monsters;

    MyPanel panel;

    MyFrame() {

        monsters = new ArrayList<Monster>();

        this.setTitle("Monster Catch Game");

        this.setSize(700, 300);

        for (int i = 0; i < 10; i++) {

            monsters.add(new Monster((int) (Math.random() \* 600), (int) (Math.random() \* 200), 1));

        }

        panel = new MyPanel();

        this.add(panel);

        this.addMouseListener(new MouseAdapter() {

            @Override

            public void mouseClicked(MouseEvent e) {

                int x = e.getX();

                int y = e.getY();

                Iterator<Monster> iterator = monsters.iterator();

                while (iterator.hasNext()) {

                    Monster m = iterator.next();

                    if (x >= m.x && x <= m.x + m.img.getWidth(null) &&

                        y >= m.y && y <= m.y + m.img.getHeight(null)) {

                        iterator.remove();

                    }

                }

                repaint();

                if (monsters.size()==0)

                    System.exit(1);

            }

        });

        Timer timer = new Timer(100, e -> {

            move();

            repaint();

        });

        timer.start();

        this.setLocationRelativeTo(null);

        this.setVisible(true);

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

    }

    void move() {

        for (Monster m : monsters) {

            m.x += m.speed;

            m.y += m.speed;

            if (m.x < 0 || m.x > getWidth() - m.img.getWidth(null)) {

                m.speed \*= -1;

            }

            if (m.y < 0 || m.y > getHeight() - m.img.getHeight(null)) {

                m.speed \*= -1;

            }

        }

    }

    class MyPanel extends JPanel {

        @Override

        protected void paintComponent(Graphics g) {

            super.paintComponent(g);

            for (Monster m : monsters) {

                m.draw(g);

            }

        }

    }

}

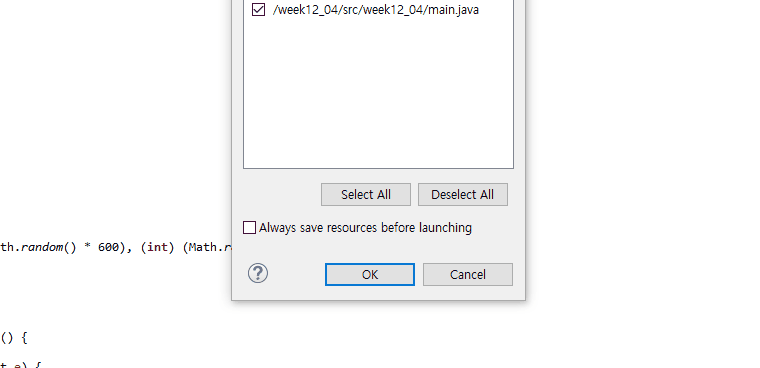
public class main {

    public static void main(String[] args) {

        MyFrame f = new MyFrame();

    }

}



이 프로그램은 10개의 몬스터를 랜덤생성하고, 몬스터를 클릭한 이후 모든 몬스터가 사라지면 자동으로 프로그램을 종료하는 기능을 한다. 우선 이 몬스터들은 monsters라는 이름의 ArrayList에 저장이 된다. for문을 통해 10개의 monster 클래스 객체를 monsters에 저장한다. 만약 클릭이 발생했을 경우, 모든 객체의 x, y값을 가져와 객체의 크기 내에 존재한다면 해당 객체를 삭제한다. 비교가 끝나면 다시 repaint를 통해 객체를 삭제한다. 만약 monsters의 크기가 0 즉슨 모든 몬스터가 사라졌다면 시스템을 종료한다.

해당 프로그램은 타이머를 통해 작동한다. 0.1초에 한번씩 작동하는 타이머를 세팅하고, 해당 타이머가 작동할 시, 객체들을 움직이도록 move를 하고, repaint를 통해 객체들을 움직이게 한다. Move에서는 몬스터들을 움직이도록 한다. 지정된 speed만큼 x와 y에 더해줌으로써 객체를 움직인다. 만약 x, y 좌표가 스크린의 영역을 벗어나면 speed에 -1을 곱해 반대방향으로 움직일 수 있도록 한다.