**package** thread;

**import** java.awt.Color;

**import** java.awt.Font;

**import** java.awt.GridLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** javax.swing.JButton;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JPanel;

**public** **class** Timer **extends** JFrame **implements** ActionListener, Runnable { // 1 ... 8 ... 11

**private** JLabel label; // 4

**private** JButton startBtn, stopBtn;

**private** **boolean** onoff; // 18 - 멈춤 버튼을 눌렀는데도 불구하고 while문에 의해 숫자가 계속 올라가는 걸 방지하기 위해

**private** **int** count = 1;

**public** Timer() {

label = **new** JLabel("0", JLabel.***CENTER***); // 5 - 필드가 널값이니 생성자를 잡자~

label.setFont(**new** Font("Arial", Font.***BOLD***, 70));

label.setForeground(Color.***RED***);

startBtn = **new** JButton("시작");

stopBtn = **new** JButton("멈춤");

JPanel p = **new** JPanel(**new** GridLayout(2, 1, 0, 5)); // 6 - 2행, 1줄, 여백, 여백의크기

p.add(startBtn);

p.add(stopBtn);

add("Center", label); // 7

add("East", p); // p패널의 동쪽에

setBounds(900, 200, 300, 200); // 2

setVisible(**true**);

setDefaultCloseOperation(***EXIT\_ON\_CLOSE***); // 종료

// 이벤트 - 클릭이벤트!

startBtn.addActionListener(**this**); // 9

stopBtn.addActionListener(**this**);

}

@Override

**public** **void** actionPerformed(ActionEvent e) { // 10

// Thread를 왜 생성해줘야 할까?

// 두가지 프로세스가 일어나기 때문.

// "시작" 혹은 "멈춤" 버튼을 누르는 것 - 1

// 버튼을 누르기 전 계속해서 숫자는 돌고있는 것 - 2

Thread t = **null**;

**if** (e.getSource() == startBtn) { // 13

t = **new** Thread(**this**); // 생성 - 익명이너가 없기 때문에 this라고 써줘도 됨

t.start(); // 스레드 시작

onoff = **true**; // 19

startBtn.setEnabled(**false**); // 15 - 스타트 버튼을 눌렀을 때 스타트 버튼 비활성화

stopBtn.setEnabled(**true**); // 대신 멈춤 버튼은 활성화

} **else** **if** (e.getSource() == stopBtn) { // 14

t = **null**; // 스레드 죽이기... 종료시켜버리기

onoff = **false**; // 20

startBtn.setEnabled(**true**); // 16 - 멈춤 버튼을 눌렀을 때 스타트 버튼 활성화

stopBtn.setEnabled(**false**); // 멈춤 버튼은 비활성화

}

}

@Override

**public** **void** run() { // 12 - Runnable의 추상메소드인 run() Override

**int** i; // 17

**for** (i = count; i <= 50; i++) {

// 만약 i = 1 이라고 해두면 멈췄다가 다시 시작할 때 멈춘 숫자에서 시작하는 것이 아닌

// 1부터 다시 시작하니까 멈춘 숫자부터 시작하도록 i = count라고 잡아줌

label.setText(i + " "); // setText는 문자열만 쓸 수 있는데 i는 int형이니까 문자열로 바꿔주기 위해 ""를 붙임

**if** (!onoff) { // 21 - 만약에 onoff에서 false가 들어오면 멈춤 버튼을 눌렀다는 거

// false의 반대인 true가 나오려면 onoff! 가 되어야 하고, true가 나오면 조건을 만족한 거니까

count = i; // i가 증가하다 중간에 끊어진 곳을 넘겨주고 break

**break**; // for문을 나가라

}

**try** {

Thread.*sleep*(100); // 천천히 올라가게끔

} **catch** (InterruptedException e) {

e.printStackTrace();

}

} // for

**if** (i > 50) { // 22 - 1 부터 50까지 다 돌았으면

count = 0; // 다시 0부터 돌아라

startBtn.setEnabled(**true**);

stopBtn.setEnabled(**false**);

}

}

**public** **static** **void** main(String[] args) {

**new** Timer(); // 3

}

}