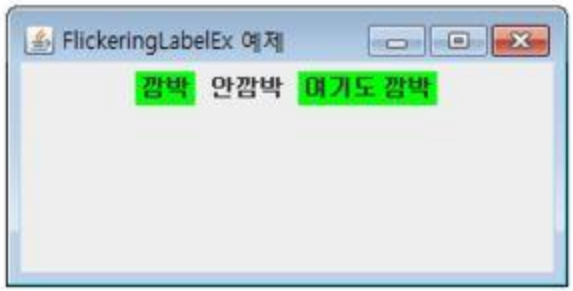
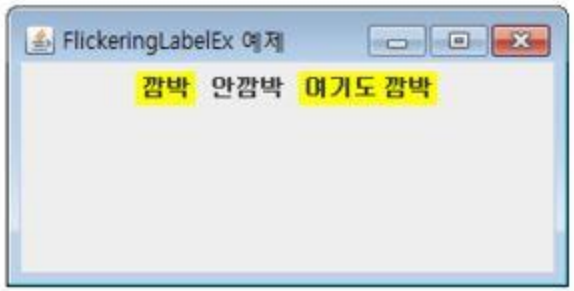
예제 13-1 : Thread를 상속받아 1초 단위의 타이머 레이블 만들기

```
import java.awt.*;
import javax.swing.*;

class TimerThread extends Thread {
    JLabel timerLabel;

    public TimerThread(JLabel timerLabel) {
        this.timerLabel = timerLabel;
    }
    public void run() {
        int n=0;

        while(true) {
            timerLabel.setText(Integer.toString(n));
            n++;
            try {
                Thread.sleep(1000);
            }
            catch(InterruptedException e) {
                return;
            }
        }
    }
}
```

```
public class ThreadTimerEx extends JFrame {
    public ThreadTimerEx() {
        setTitle("ThreadTimerEx 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        JLabel timerLabel = new JLabel();
        timerLabel.setFont(new Font("Gothic", Font.ITALIC, 80));

        TimerThread th = new TimerThread(timerLabel);
        c.add(timerLabel);

        setSize(300,150);
        setVisible(true);

        th.start();
    }

    public static void main(String[] args) {
        new ThreadTimerEx();
    }
}
```

예제 실행 : 깜박이는 레이블 만들기





예제 13-3 : 깜박이는 문자열을 가진 레이블 만들기

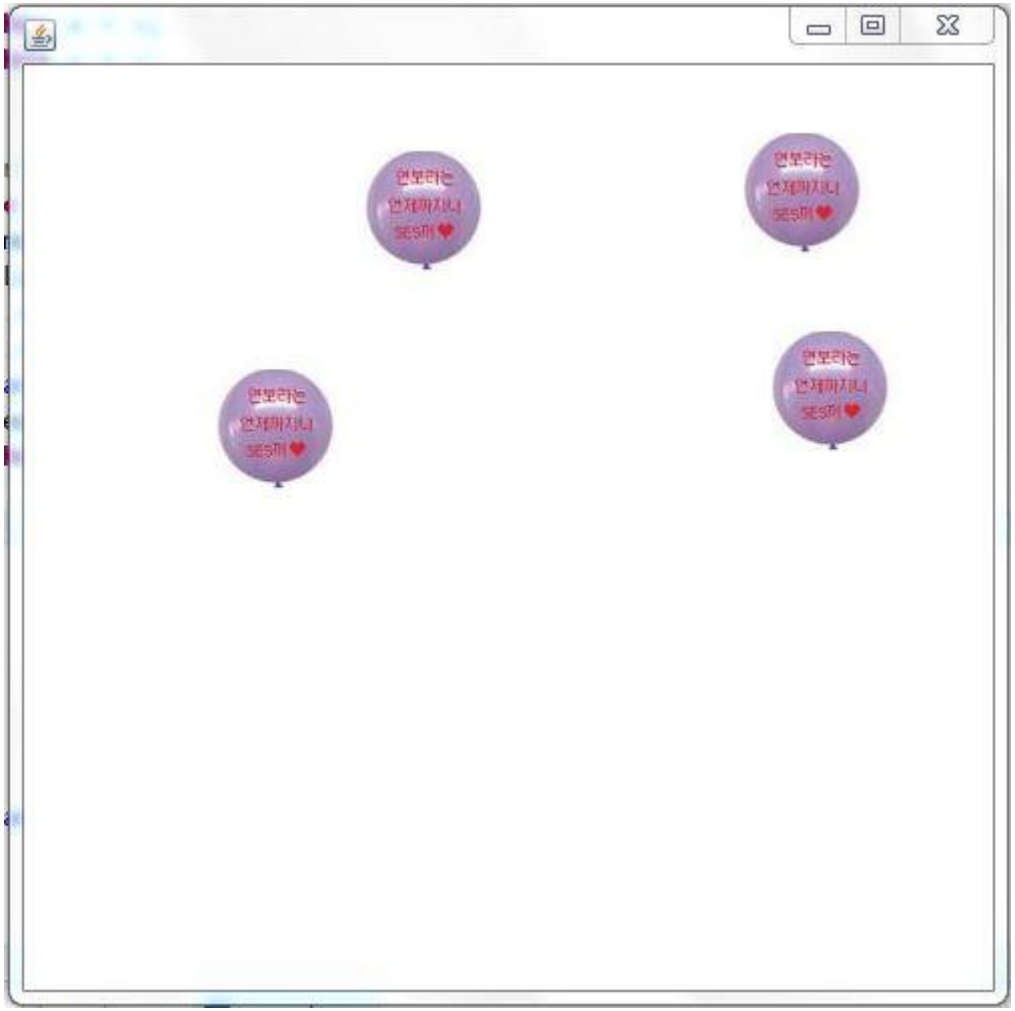

```
import java.awt.*;
import javax.swing.*;

class FlickeringLabel extends JLabel implements
Runnable{
    public FlickeringLabel(String text) {
        super(text); // JLabel 생성자 호출
        setOpaque(true); // 배경색 변경이 가능하도록 설정

        Thread th = new Thread(this);
        th.start();
    }

    public void run() {
        int n=0;
        while(true) {
            if(n == 0)
                setBackground(Color.YELLOW);
            else
                setBackground(Color.GREEN);
            if(n == 0) n = 1;
            else n = 0;
            try {
                Thread.sleep(500); // 0.5초 동안 잠을 잔다.
            }
            catch(InterruptedException e) {
                return;
            }
        }
    }
}
```

```
public class FlickeringLabelEx extends JFrame {
    public FlickeringLabelEx() {
        setTitle("FlickeringLabelEx 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        // 깜박이는 레이블 생성
        FlickeringLabel fLabel = new FlickeringLabel("깜박");

        // 깜박이지 않는 레이블 생성
        JLabel label = new JLabel("안깜박");

        // 깜박이는 레이블 생성
        FlickeringLabel fLabel2 = new FlickeringLabel("여기도 깜박");

        c.add(fLabel);
        c.add(label);
        c.add(fLabel2);

        setSize(300,150);
        setVisible(true);
    }

    public static void main(String[] args) {
        new FlickeringLabelEx();
    }
}
```

<연습문제> 한국의 인구수 증가를 스레드를 통해 표현되도록 프로그래밍 해보자

전체 인구수를 51712221 명이라고 가정하고
초당 2명씩 증가 한다고 가정한다.

<예제> 프레임 위에 마우스를 클릭하면 풍선이 만들어져 올라가는 프로그램을 작성하라. (풍선은 50ms마다 5 pixcel 위로 올라간다. 풍선은 프레임을 벗어나면 삭제된다)



1. JFrame을 상속받는다. (크기는 500*500)
2. JPanel을 contentPane에 붙인다. (JPanel의 레이아웃 관리자는 null값을 가진다)
3. JPanel에 MouseAdapter를 등록한다.
4. MouseAdapter를 상속받는 클래스(MyMouseAdapter)를 만든다.
5. MouseAdapter의 mousePressed(MouseEvent e)를 메소드 오버라이딩한다
6. 마우스를 Pressed했을때 스레드 객체를 생성하고 작동 (start())시킨다.
   (마우스 pressed에 동작하는 스레드 클래스는 아래와 같이 정의한다.)
7. Thread를 상속받는 BallonThread 클래스를 만든다.
8. run() 메소드에 풍선이미지를 가진 Jlabel을 만든다.
9. JPanel의 레이아웃은 배치관리자를 갖지 않음으로 JLabel의 setLocation(), setSize()를설정하여 panel에 붙여준다. 위치값은 마우스이벤트가 발생한 좌표값을 넣어준다. (e.getX(), e.getY())
   repaint()메소드를 통해 풍선이미지의 레이블을 화면에 그려준다.
   이 레이블은 50ms마다 이동함으로 Thread.sleep(50)을 통해 쉬어준후 좌표값을 이동시킨다. 풍선이 위로 올라감으로 y좌표값을 5픽셀씩 감소시키면서(y-=20)
   화면에 이미지레이블을 그려준다.(repaint())
   화면의 가장 윗부분은 y값이 0이다. y값이 0보다 작으면 panel안의 component인 이미지 레이블(Jlabel)을 삭제한다. (panel.remove(레이블객체))

10. main() 메소드에 클레스 ThreadBalloonEx 객체를 생성한다.

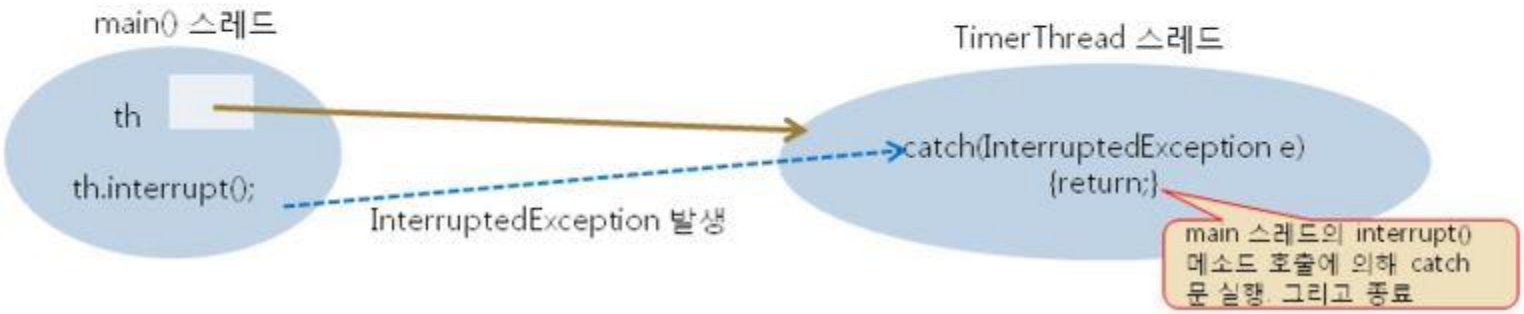
스레드 종료와 타 스레드 강제 종료

- 스스로 종료
  - run() 메소드 리턴
- 타 스레드에서 강제 종료 : interrupt() 메소드 사용

```
public static void main(String [] args) {
    TimerThread th = new  TimerThread();
    th.start();

    th.interrupt(); // TimerThread 강제 종료
}
```

```
class TimerThread extends Thread {
    int n = 0;
    public void run() {
        while(true) {
            System.out.println(n); // 화면에 카운트 값 출력
            n++;
            try {
                sleep(1000);
            }
            catch(InterruptedException e){
                return; // 예외를 받고 스스로 리턴하여 종료
            }
        }
    }
}
```

만일 return 하지 않으면 스레드는 종료하지 않음



예제 13-4 : 타이머 스레드 강제 종료


```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TimerRunnable implements Runnable {
    JLabel timerLabel;

    public TimerRunnable(JLabel timerLabel) {
        this.timerLabel = timerLabel;
    }

    public void run() {
        int n=0;
        while(true) {
            timerLabel.setText(Integer.toString(n));
            n++;
            try {
                Thread.sleep(1000); // 1초 동안 잠을 잔다.
            }
            catch(InterruptedException e) {
                return; // 예외가 발생하면 스레드 종료
            }
        }
    }
}
```

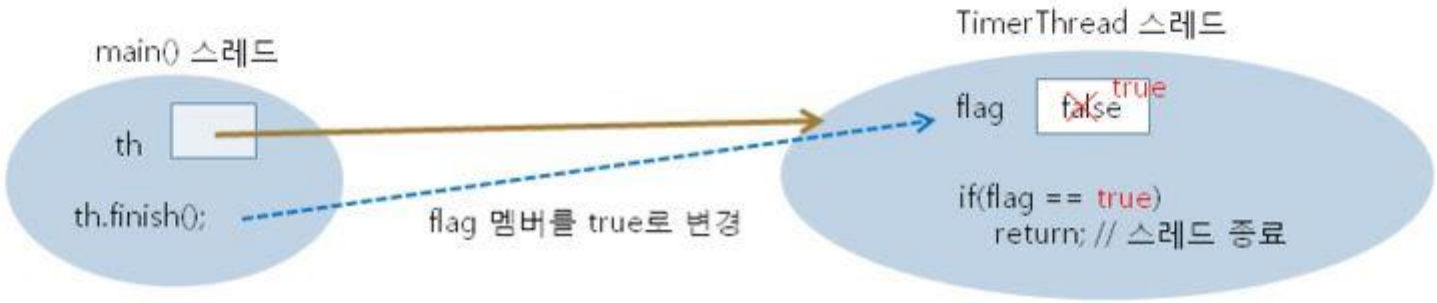
```
public class ThreadInterruptEx extends JFrame {
    Thread th;

    public ThreadInterruptEx() {
        setTitle("hreadInterruptEx 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        JLabel timerLabel = new JLabel();
        timerLabel.setFont(new Font("Gothic", Font.ITALIC, 80));

        TimerRunnable runnable = new TimerRunnable(timerLabel);
        th = new Thread(runnable); // 스레드 생성
        c.add(timerLabel);

        // 버튼을 생성하고 Action 리스너 등록
        JButton btn =new JButton("kill Timer");
        btn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                th.interrupt(); // 타이머 스레드 강제 종료
                JButton btn = (JButton)e.getSource();
                btn.setEnabled(false); // 버튼 비활성화
            }
        });
        c.add(btn);
        setSize(300,150);
        setVisible(true);

        th.start(); // 스레드 동작시킴
    }
    public static void main(String[] args) {
        new ThreadInterruptEx();
    }
}
```

flag를 이용한 종료

- 스레드 A가 스레드 B의 flag를 true로 만들면, 스레드 B가 스스로 종료하는 방식

```
class TimerThread extends Thread {
    int n = 0;
    bool flag = false; // false로 초기화
    public void finish() { flag = true; }
    public void run() {
        while(true) {
            System.out.println(n); // 화면에 카운트 값 출력
            n++;
            try {
                sleep(1000);
                if(flag == true)
                    return; // 스레드 종료
            }
            catch(InterruptedException e){
                return;
            }
        }
    }
}
```

```
public static void main(String [] args) {
    TimerThread th = new  TimerThread();
    th.start();

    th.finish(); // TimerThread 강제 종료
}
```



예제 실행 결과

스레드가 작동함

스레드가 종료하였음

컨텐트팬에 마우스를 클릭하면 스레드 종료

예제 13-5
flag를 이용한 스레드 강제 종료

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class RandomThread extends Thread {
    Container contentPane;
    boolean flag=false; // 스레드의 종료 명령을 표시하는 플래그.
                        // true : 종료 지시

    public RandomThread(Container contentPane) {
        this.contentPane = contentPane;
    }

    void finish() { // 스레드 종료 명령을 flag에 표시
        flag = true;
    }

    public void run() {
        while(true) {
            int x = ((int)(Math.random()*contentPane.getWidth()));
            int y = ((int)(Math.random()*contentPane.getHeight()));
            JLabel label = new JLabel("Java"); //새 레이블 생성
            label.setSize(80, 30);
            label.setLocation(x, y);
            contentPane.add(label);
            contentPane.repaint();
            try {
                Thread.sleep(300); // 0.3초 동안 잠을 잔다.
                if(flag==true) {
                    contentPane.removeAll();
                    label = new JLabel("finish");
                    label.setSize(80, 30);
                    label.setLocation(100, 100);
                    label.setForeground(Color.RED);
                    contentPane.add(label);
                    contentPane.repaint();
                    return; // 스레드 종료
                }
            }
            catch(InterruptedException e) {return;    }
        }
    }
}
```

```
public class ThreadFinishFlagEx extends JFrame {
    RandomThread th; // 스레드 레퍼런스

    public ThreadFinishFlagEx() {
        setTitle("ThreadFinishFlagEx 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(null);

        c.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent e) {
                th.finish(); // RandomThread 스레드 종료 명령
            }
        });
        setSize(300,200);
        setVisible(true);

        th = new RandomThread(c); // 스레드 생성
        th.start(); // 스레드 동작시킴
    }

    public static void main(String[] args) {
        new ThreadFinishFlagEx();
    }
}
```