

**CSE2016**

**프로그램설계방법론**

# **GUI와 이벤트 구동 프로그래밍**

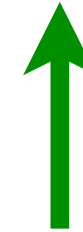
## **Graphical User Interface and Event-Driven Programming**

**도경구**

한양대학교 ERICA 소프트웨어학부



컴퓨터와 사용자 사이의 소통 창구



# Graphical User Interface



GUI

문자가 아닌 눈에 보이는 다른 것들로 소통

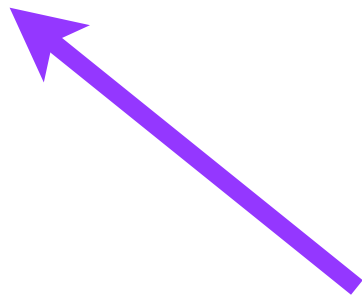
java.awt  
javax.swing

AWT = Abstract Window Toolkit

사용자 주도로  
수시로 발생하는 일

### Action Event

- 마우스 움직임
- 버튼 누름
- 메뉴 선택
- 키보드 입력



# Event-Driven Programming

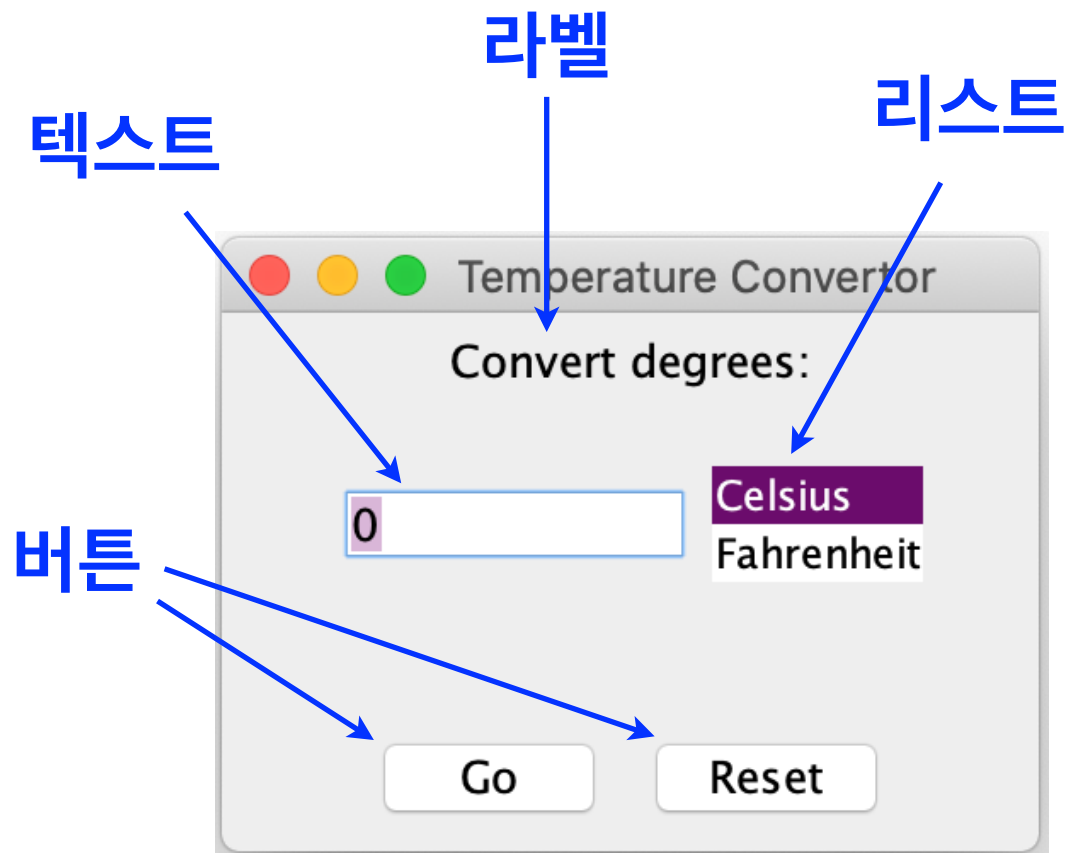
Event-handler = Action Listener

액션 이벤트가 발생하기를 기다리고 있다가,  
액션 이벤트가 발생하면 처리

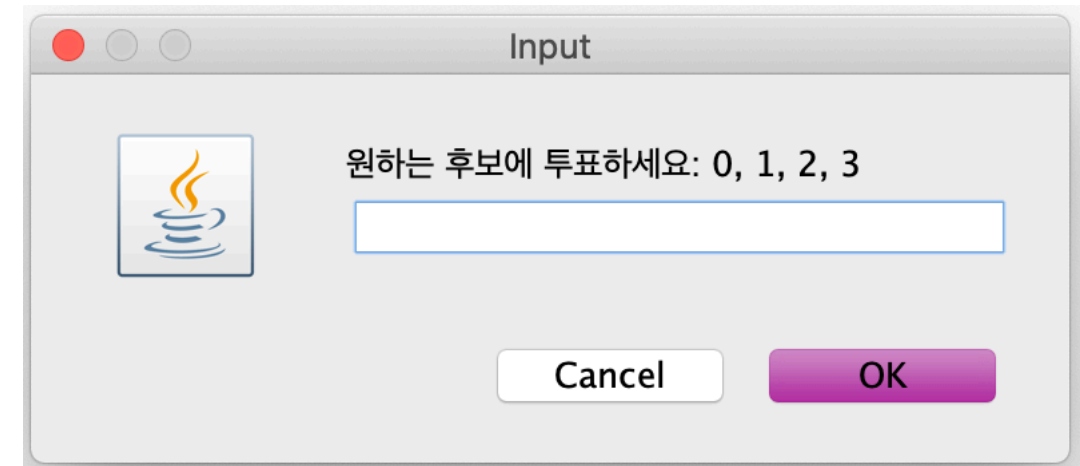
## The AWT/Swing Terminology

용어		기능
component	컴포넌트	이벤트가 발생하는 객체로서 <u>크기</u> 와 <u>위치</u> 가 있음 예: 라벨, 텍스트, 버튼, 리스트 등.
container	컨테이너	다른 컴포넌트를 1개 이상 담을 수 있는 컴포넌트
panel	패널	표준 컨테이너
window	윈도우	화면에 보이는 가장 바깥에 위치한 컨테이너로서 패널을 담고 있음
frame	프레임	윈도우에 제목과 메뉴가 달린 틀
dialog	대화창	대화를 위하여 띄우는 임시 창
menu bar	메뉴바	선택할 메뉴가 있으며 프레임에 달려 있음

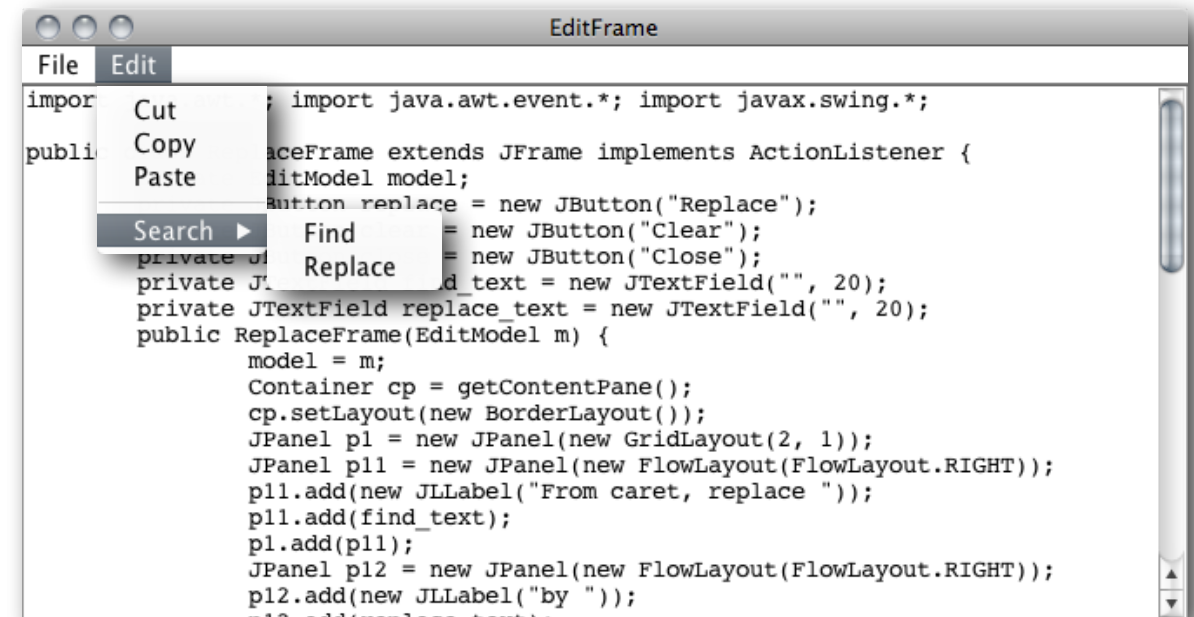
## 윈도우 프레임 window



## 대화창 dialog



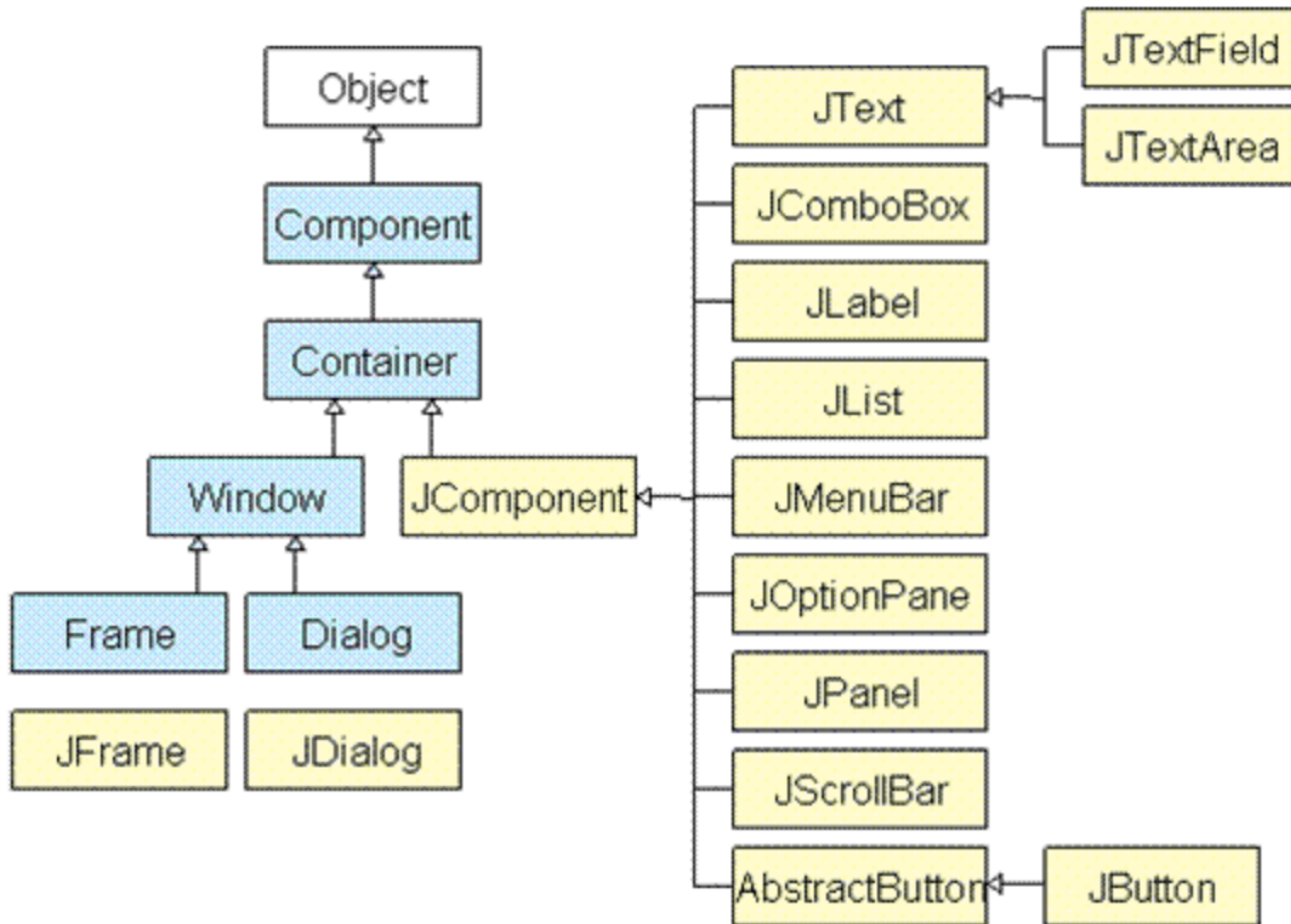
## 메뉴바 menu bar



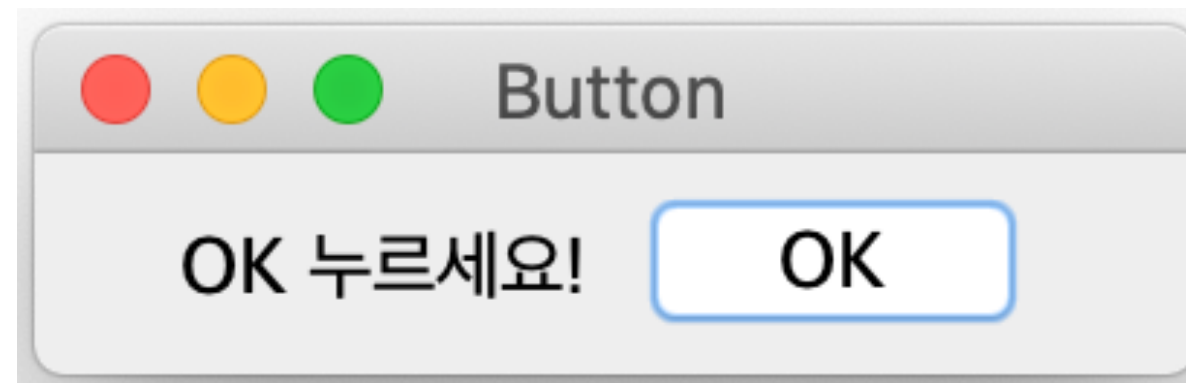
## 자리 나누기 Layout

1. Flow layout : 나란히 가로로 나열
2. Border layout : 동, 서, 남, 북, 가운데로 나누어 자리 배정
3. Grid layout : 가로, 세로 균일한 크기로 바둑판 모양으로 자리 배정

# The AWT/Swing Class Hierarchy



## 사례#1 : 라벨, 버튼 프레임을 윈도우에 배치하기

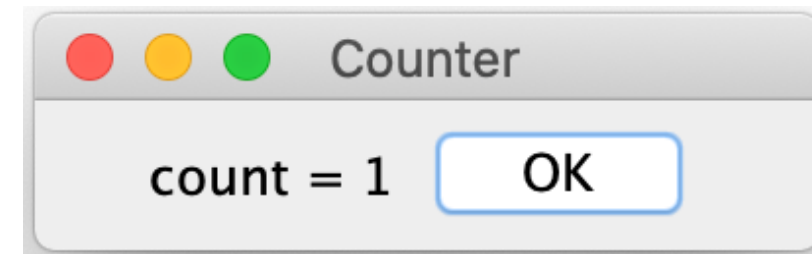
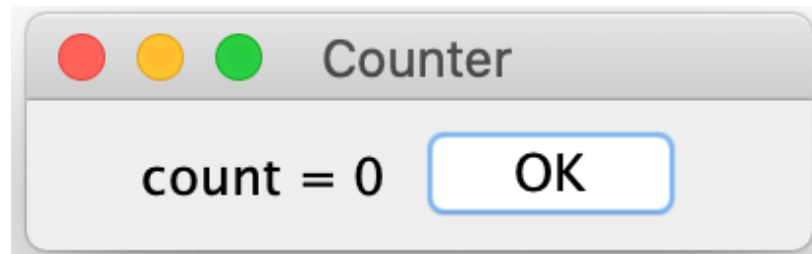


**JFrame 상속**

**코딩 따라 하기**

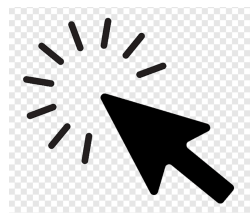
## 사례#2 : 카운터 만들기

버튼을 누를 때마다 1씩 증가하는 카운터 만들기



```
public interface ActionListener {
    /** actionPerformed - 액션 이벤트를 처리
     * @param e - 발생한 이벤트 관련 정보 */
    public void actionPerformed(ActionEvent e);
}
```

```
public class C implements ActionListener {
    . . .
    public void actionPerformed(ActionEvent e) {
        . . .
    }
}
```



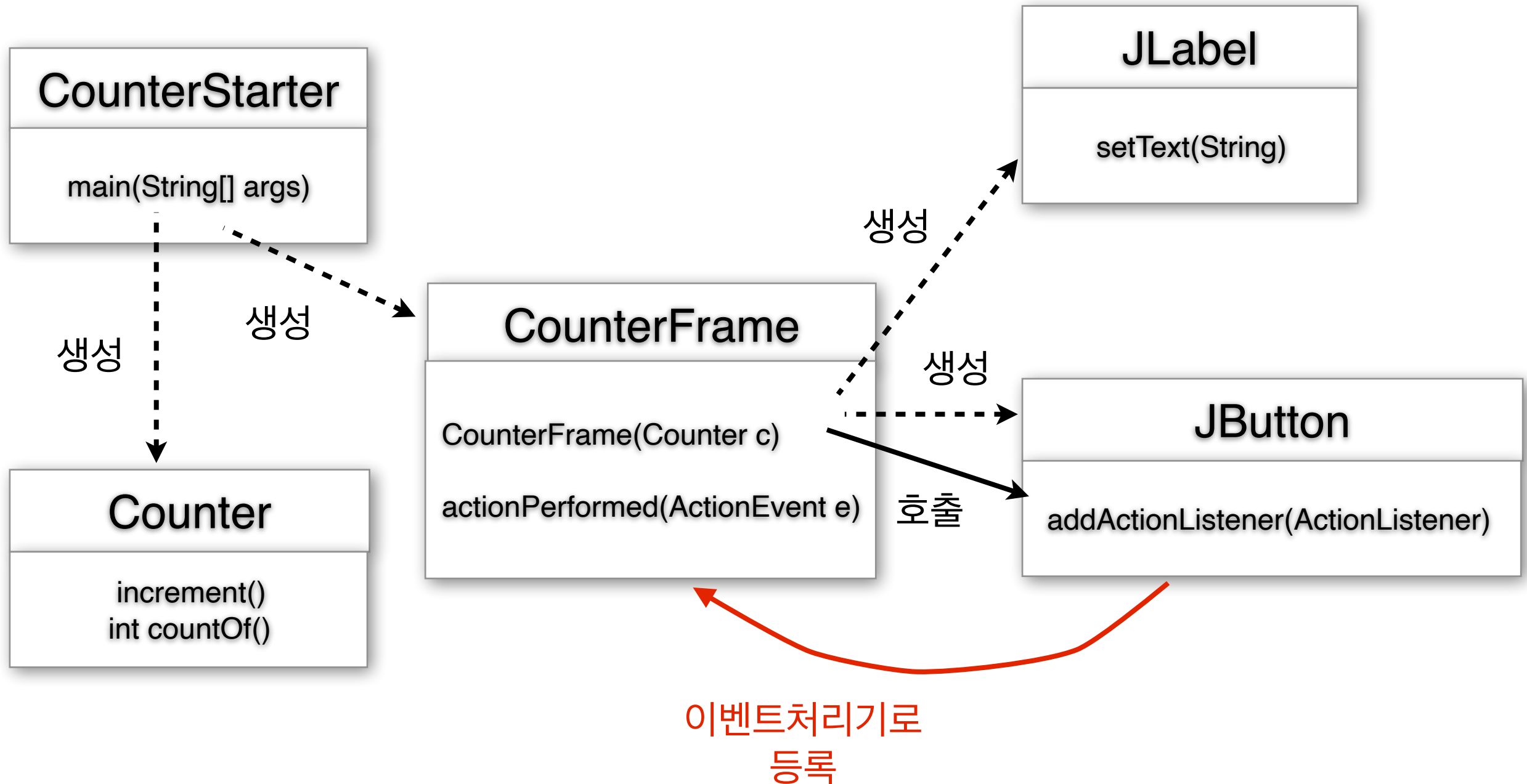
**ActionEvent**  
"button click"

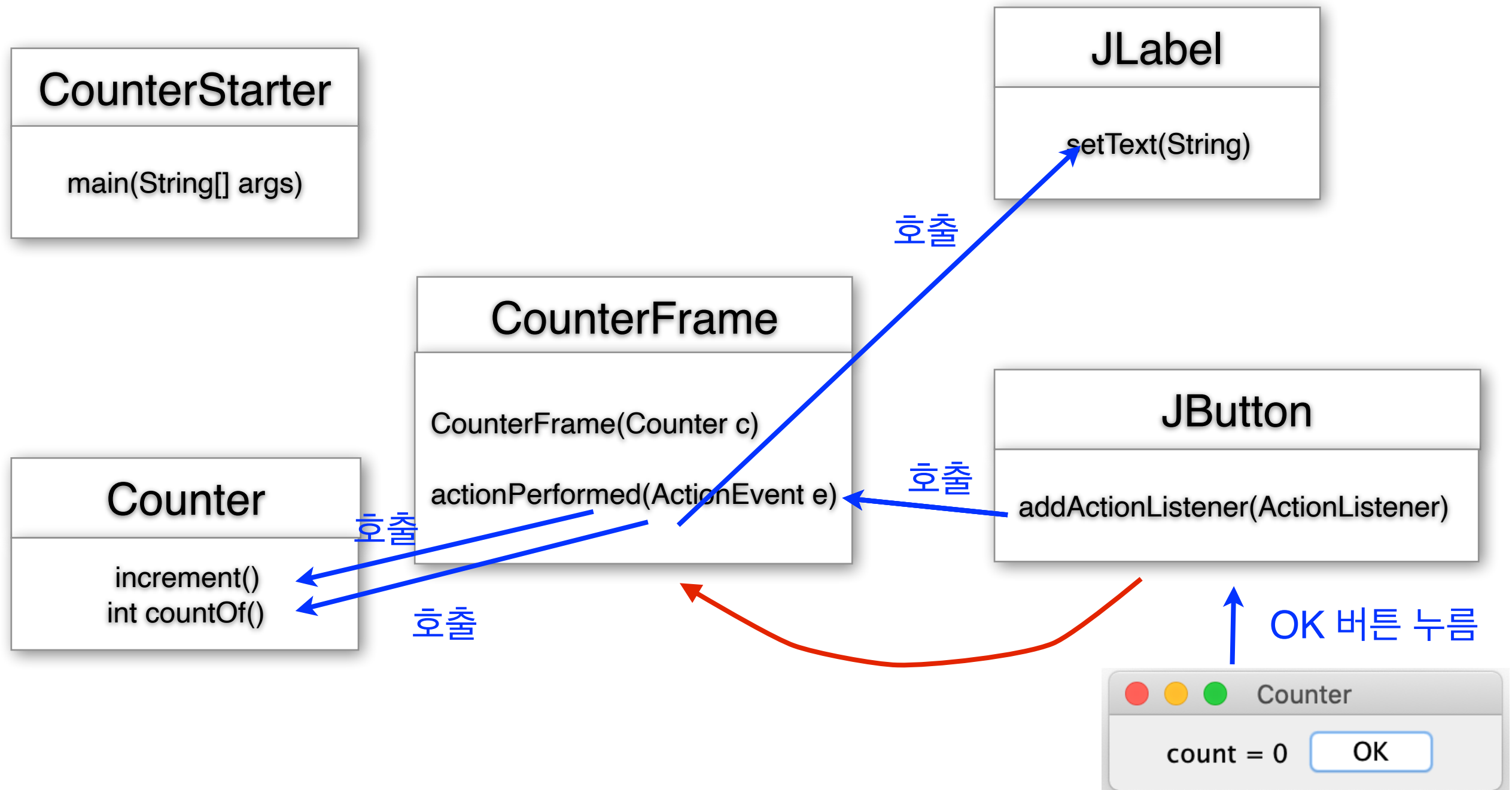
actionPerformed

**ActionListener**



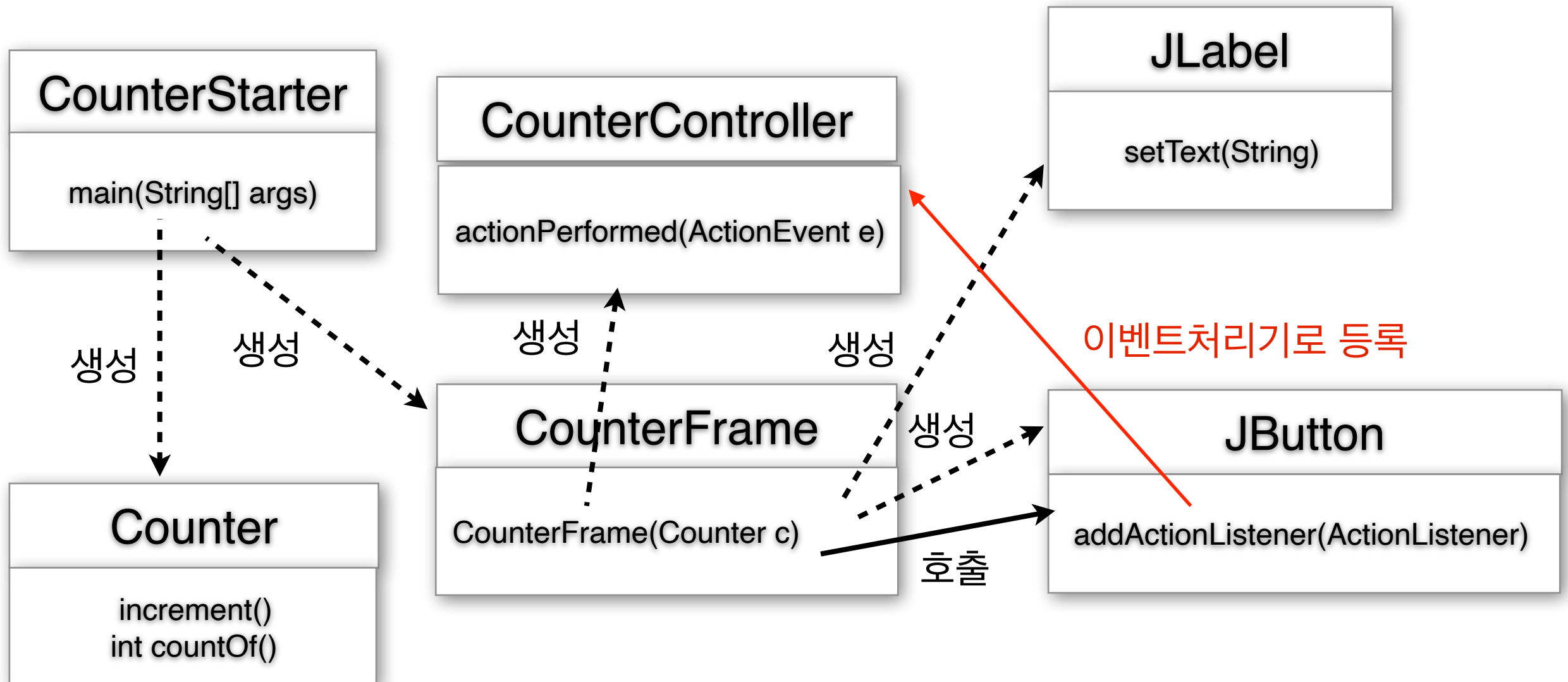
## 1 단계



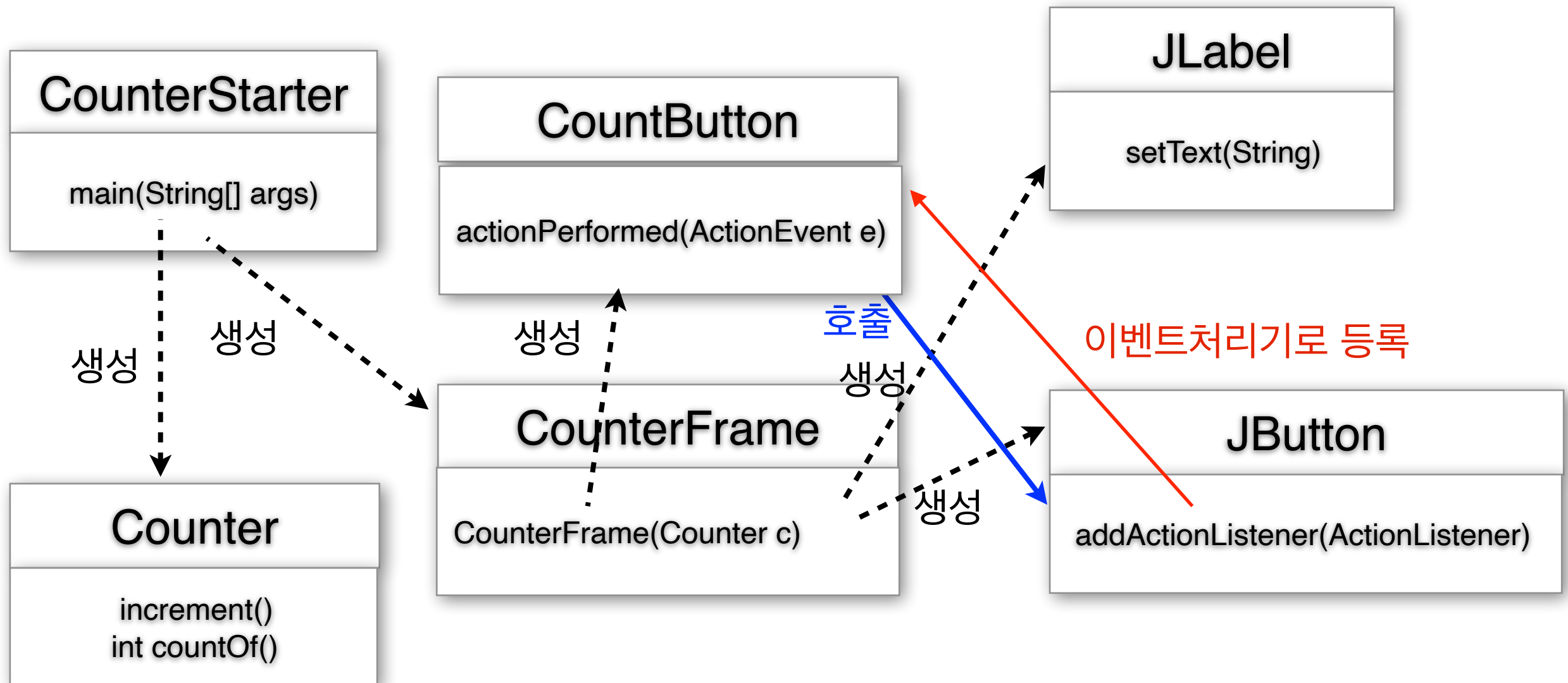


코딩 따라 하기

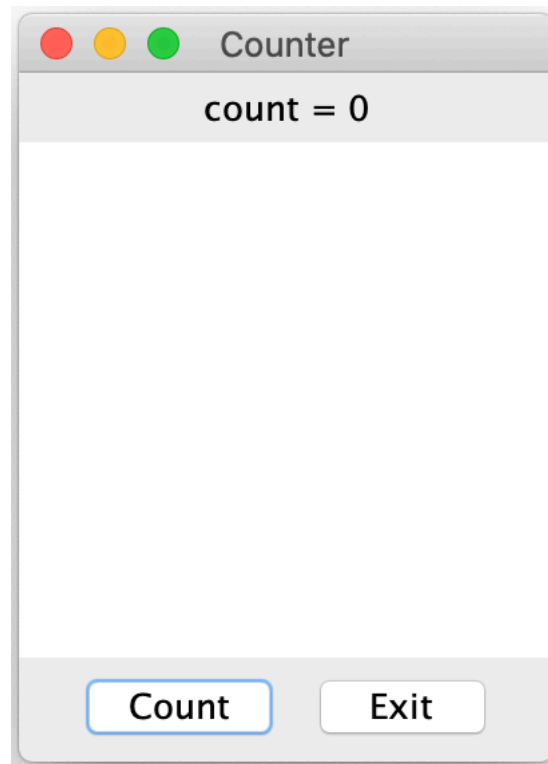
## 2 단계 - 컨트롤러와 뷰 분리



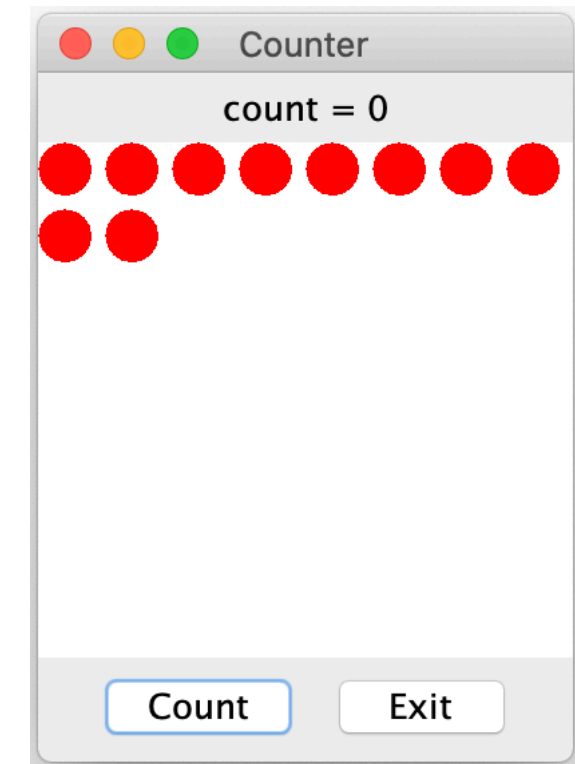
## 3 단계 - 버튼 전용 컨트롤러



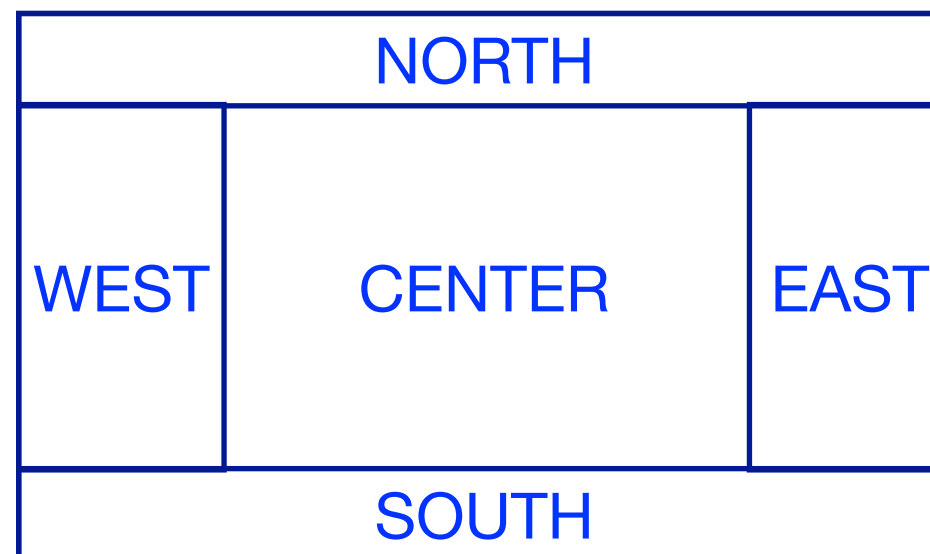
## 4 단계 - 그래픽 카운터, 구역 정리, 버튼 추가



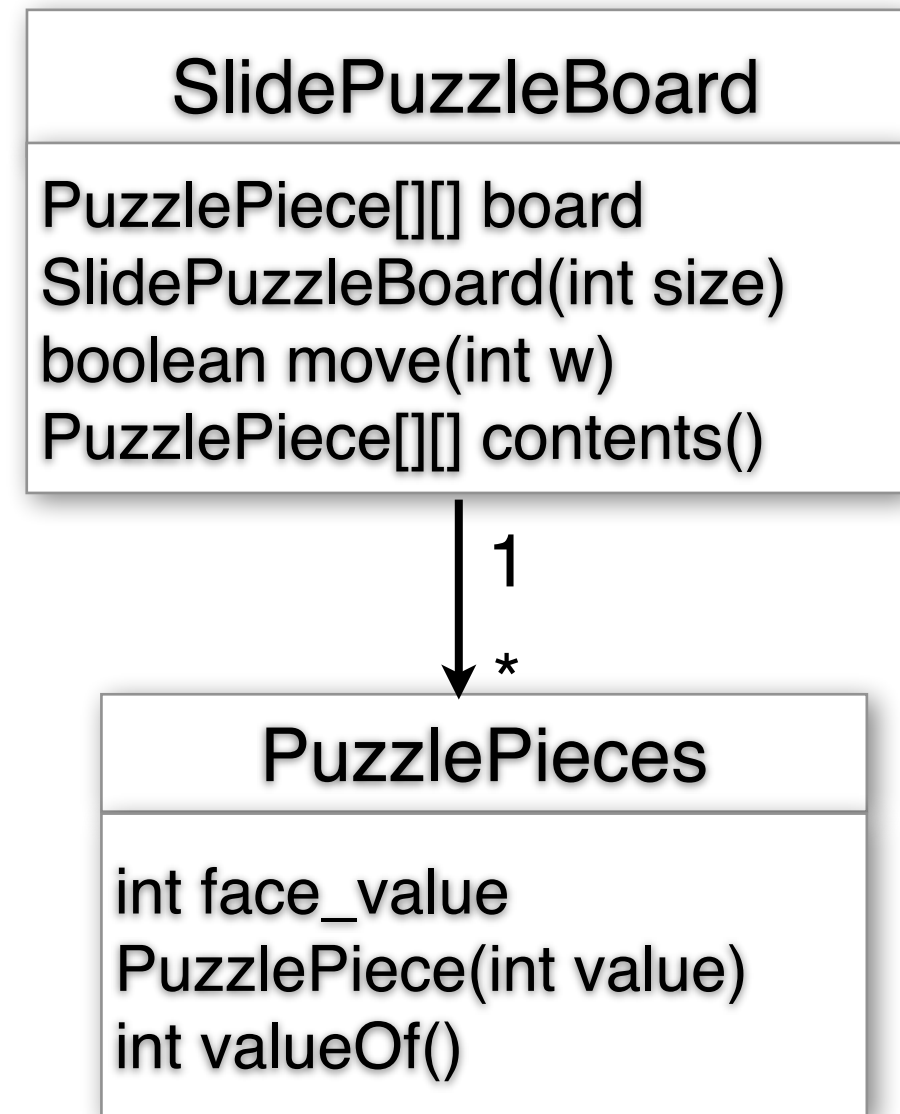
...



### Border Layout

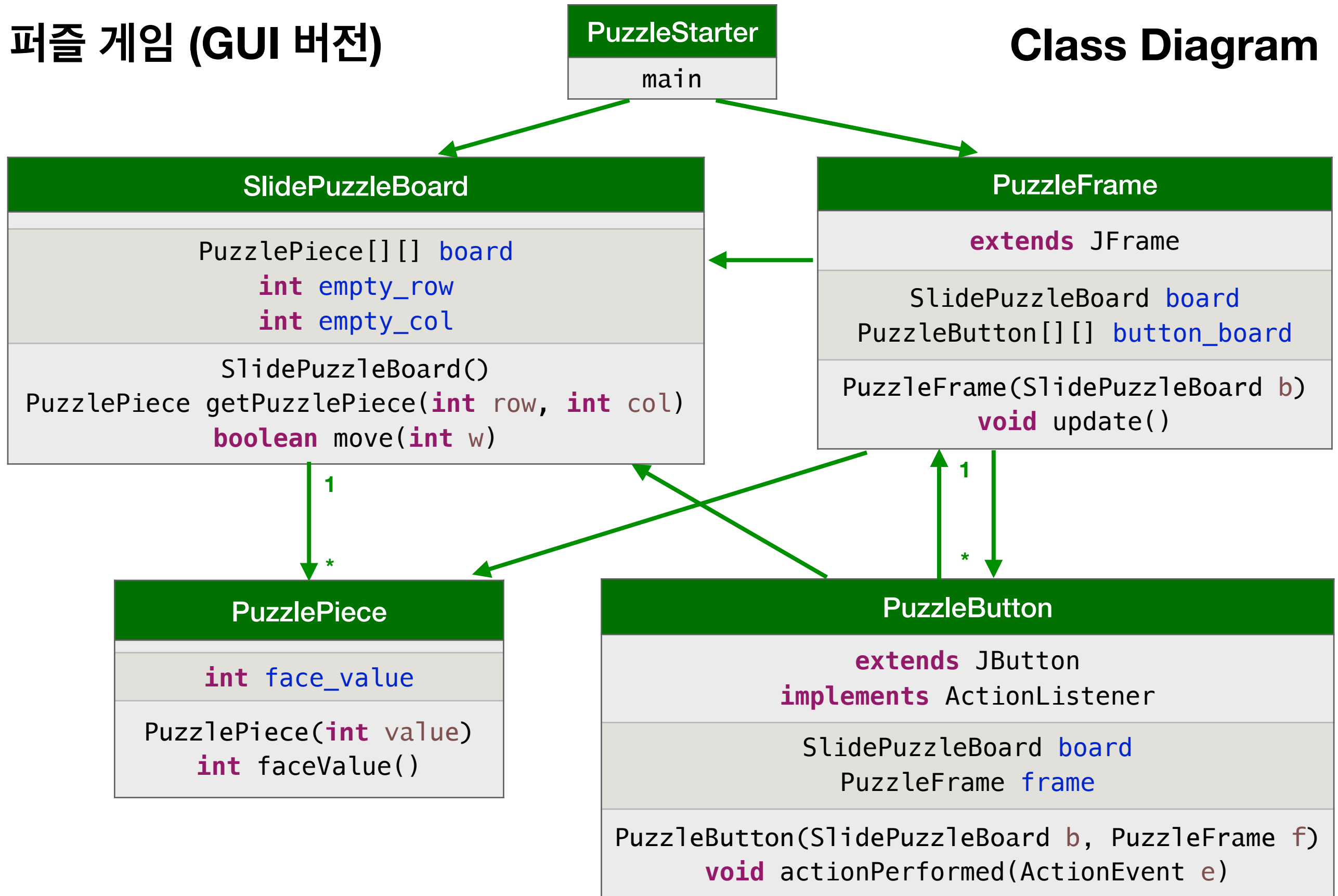


## Lab#1 - 퍼즐 게임 (GUI 버전)



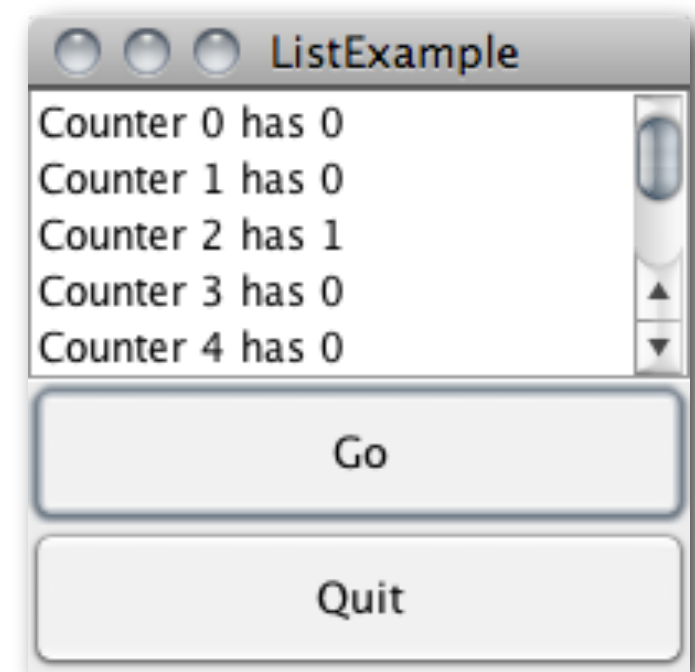
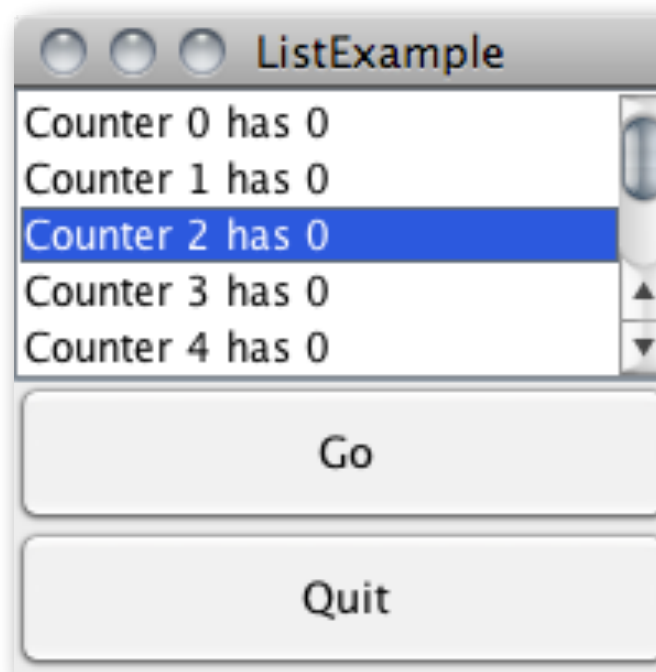
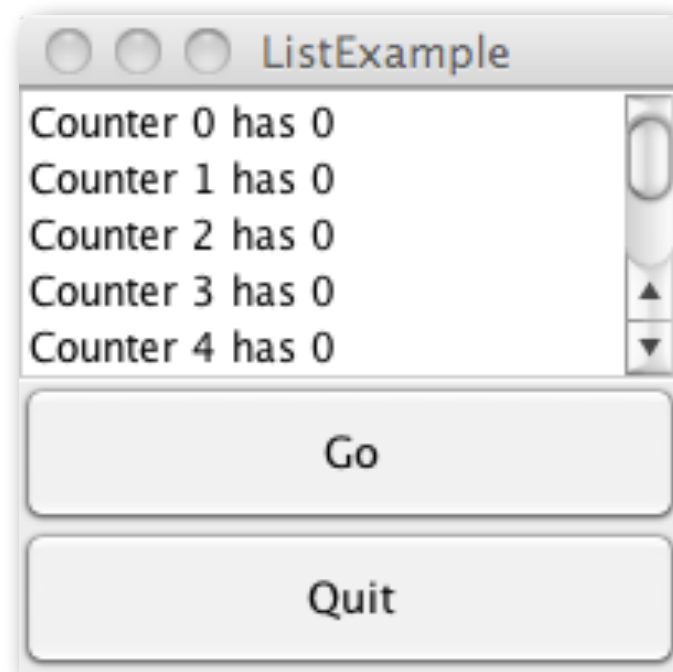
## 퍼즐 게임 (GUI 버전)

## Class Diagram



## Lab#2 - 스크롤 리스트

### List Selection Event

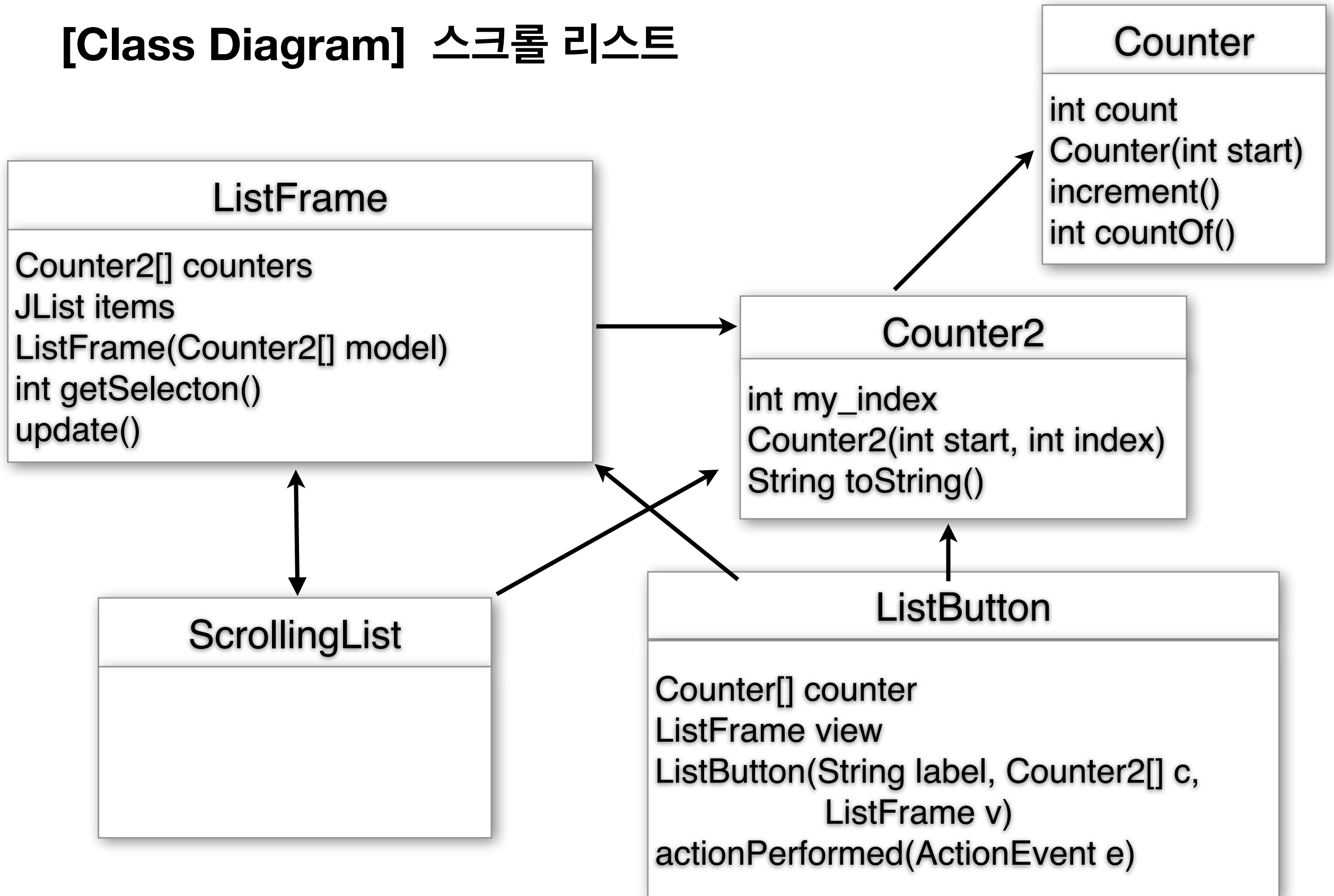


scrollinglist

코드 읽기



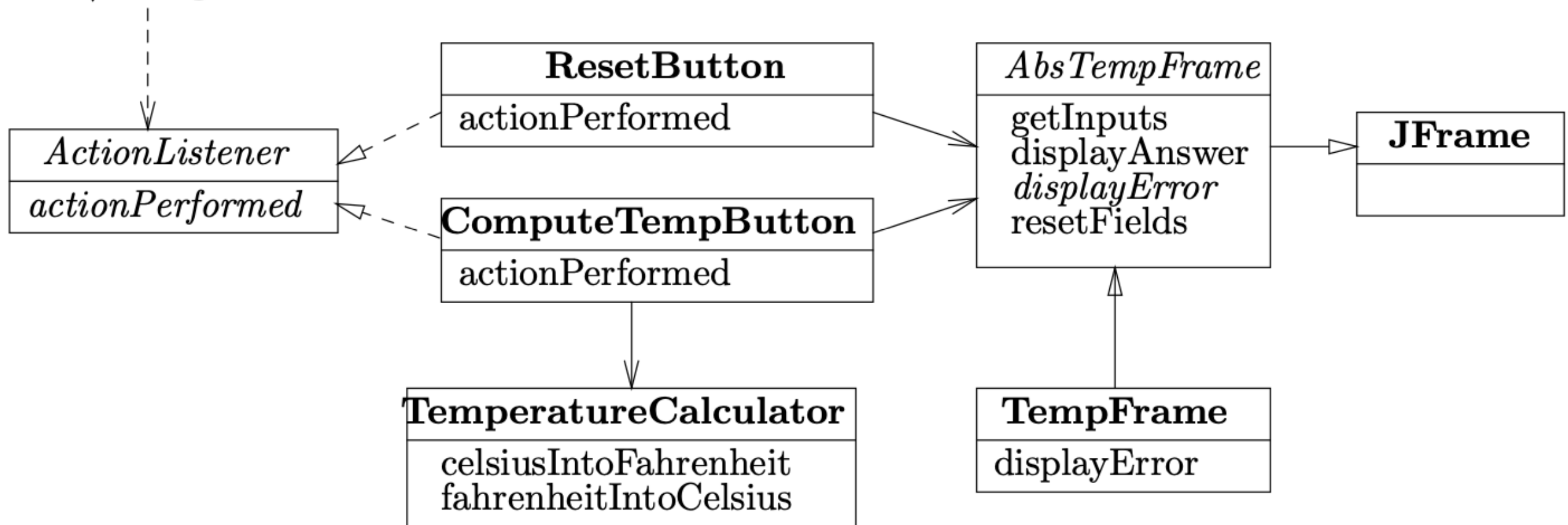
## [Class Diagram] 스크롤 리스트



코딩 따라 읽기

## Lab#3 - 텍스트 필드

AWT/Swing classes that detect events

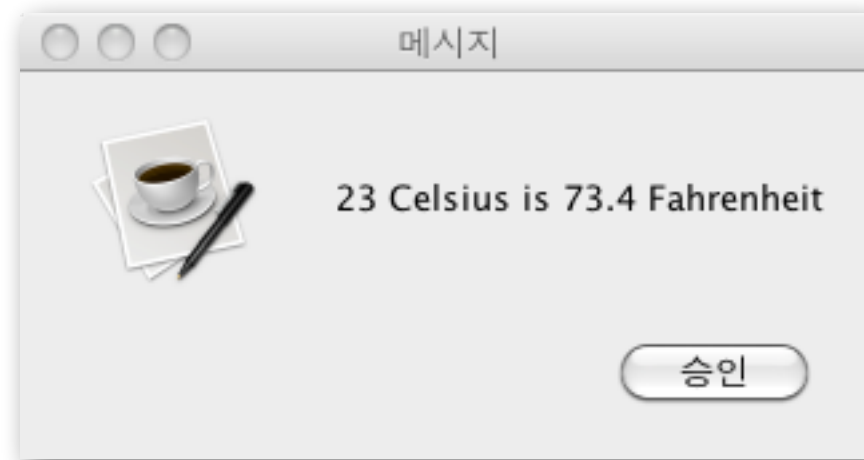


tempconv

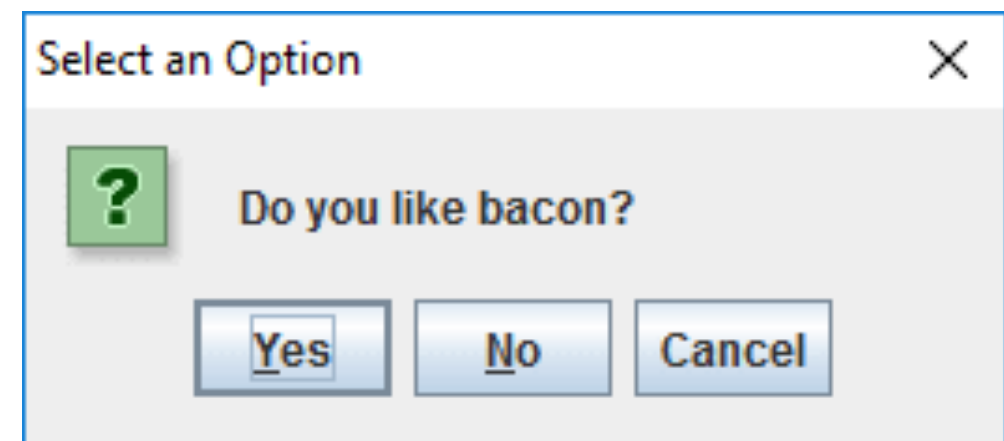
코드 읽기

# Dialogs

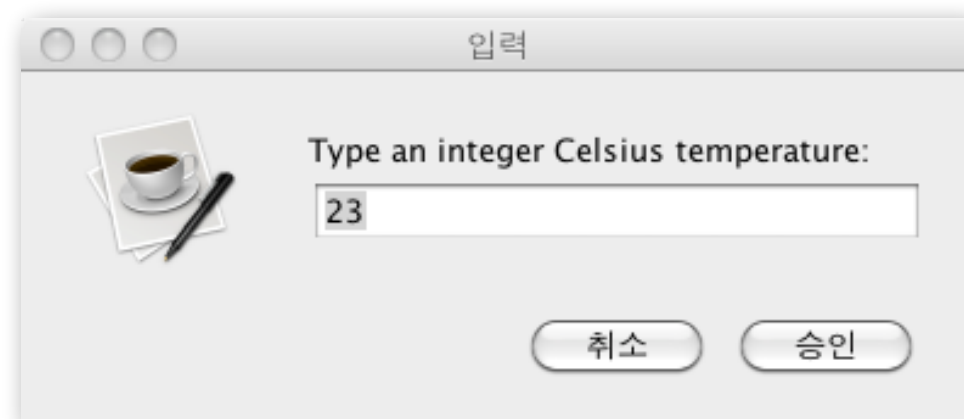
- message dialog



- confirm dialog

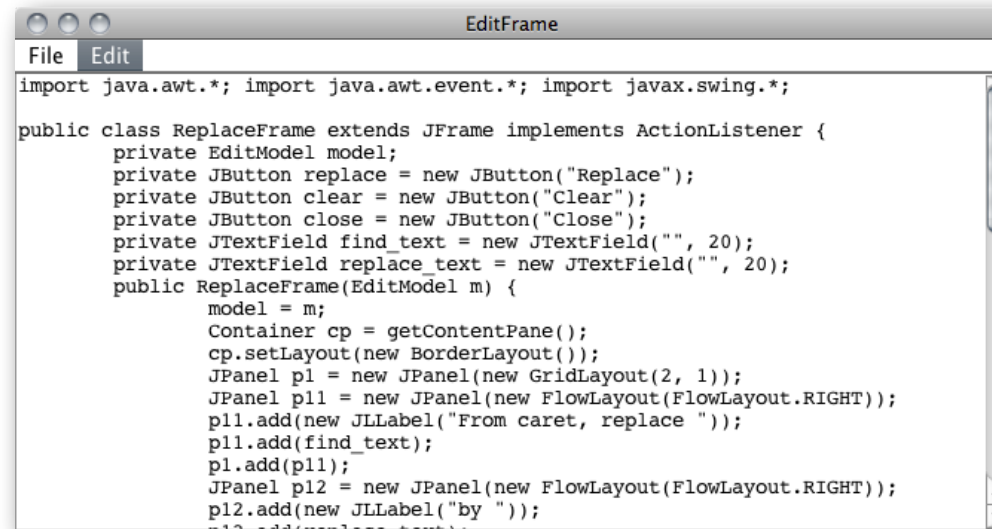


- input dialog



# Text Area

텍스트를 여러 줄 입력할 수 있는 텍스트 컴포넌트



```

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

public class ReplaceFrame extends JFrame implements ActionListener {
    private EditModel model;
    private JButton replace = new JButton("Replace");
    private JButton clear = new JButton("Clear");
    private JButton close = new JButton("Close");
    private JTextField find_text = new JTextField("", 20);
    private JTextField replace_text = new JTextField("", 20);
    public ReplaceFrame(EditModel m) {
        model = m;
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        JPanel p1 = new JPanel(new GridLayout(2, 1));
        JPanel p11 = new JPanel(new FlowLayout(FlowLayout.RIGHT));
        p11.add(new JLabel("From caret, replace "));
        p11.add(find_text);
        p1.add(p11);
        JPanel p12 = new JPanel(new FlowLayout(FlowLayout.RIGHT));
        p12.add(new JLabel("by "));
        p12.add(replace_text);
    }
}

```

```
Container cp = getContentPane();
```

```
. . .
```

```
JTextArea text = new JTextArea("", 20, 40);
```

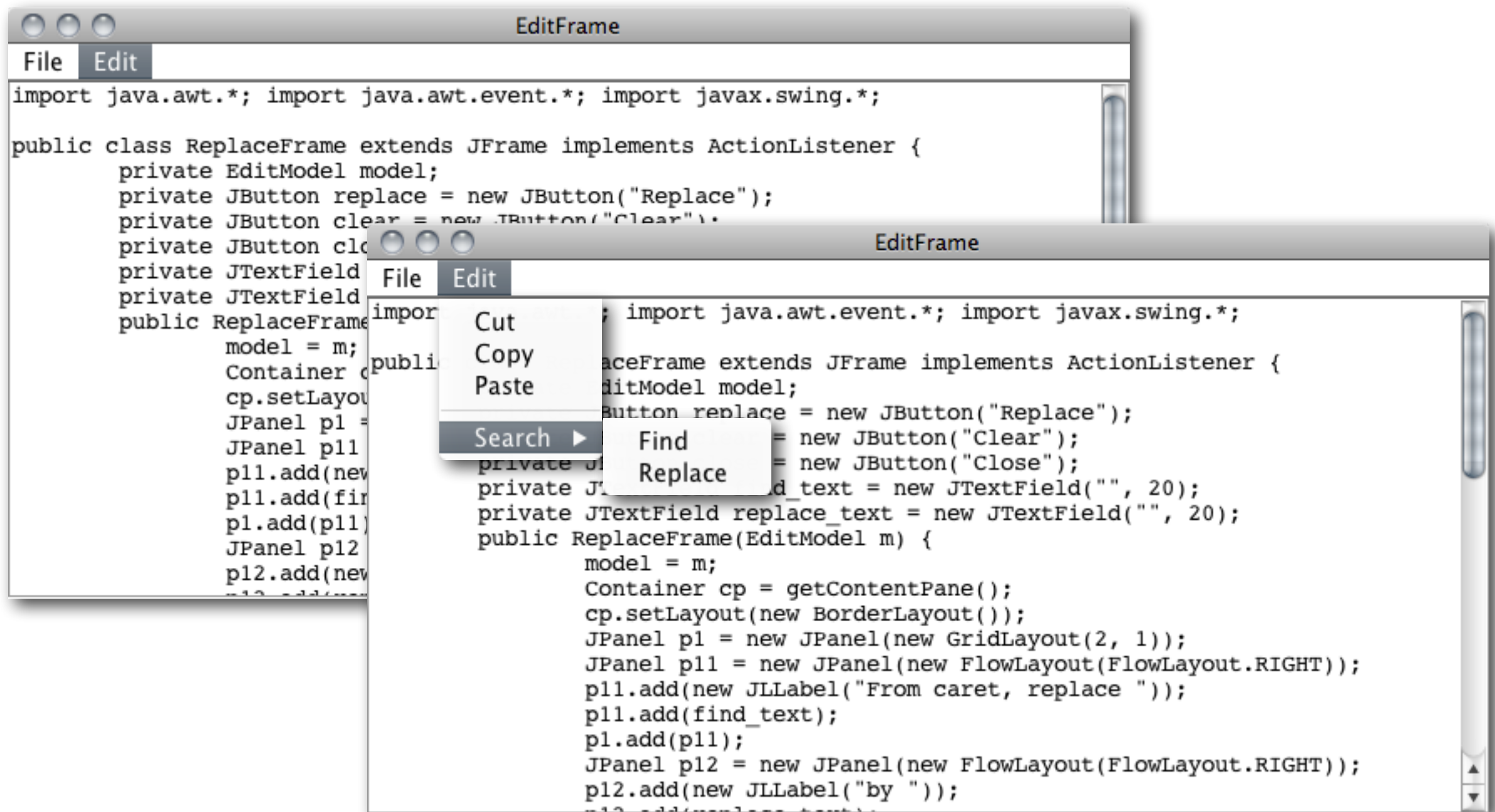
```
text.setLineWrap(true);
```

```
text.setFont(new Font("Courier", Font.PLAIN, 14));
```

```
JScrollPane sp = new JScrollPane(text)
```

```
cp.add(sp);
```

## Lab#4 - 텍스트 편집기



texteditor

코드 읽기