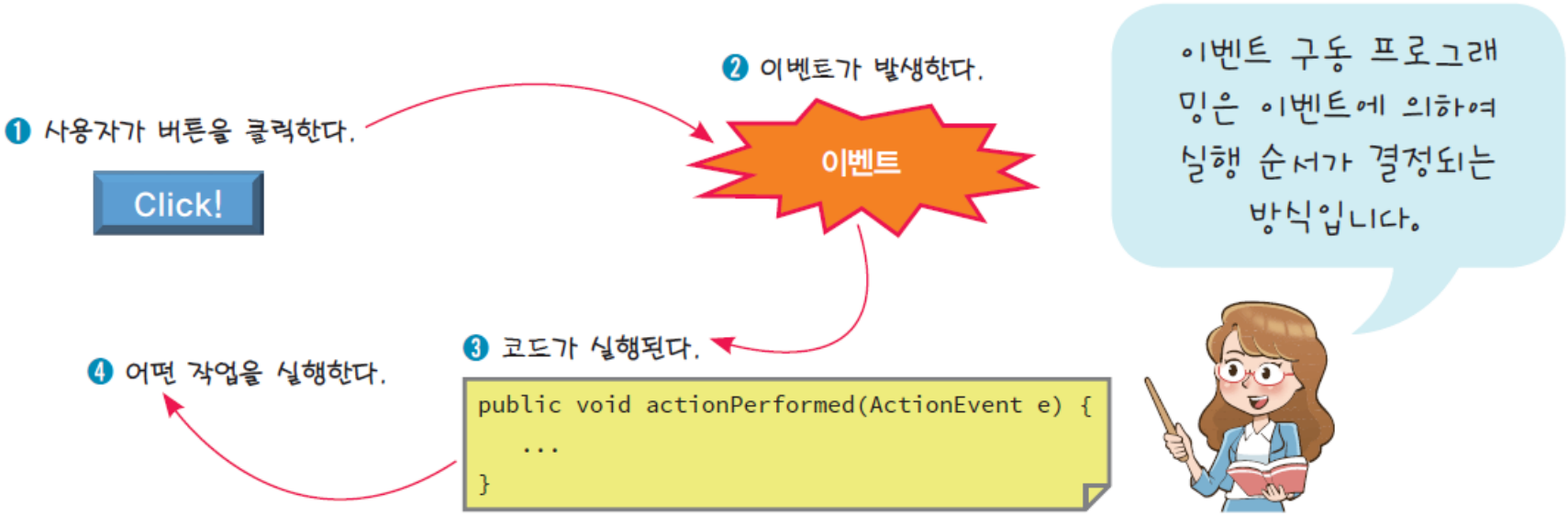
# 13. 이벤트처리

강동기

# 이벤트 구동 프로그래밍

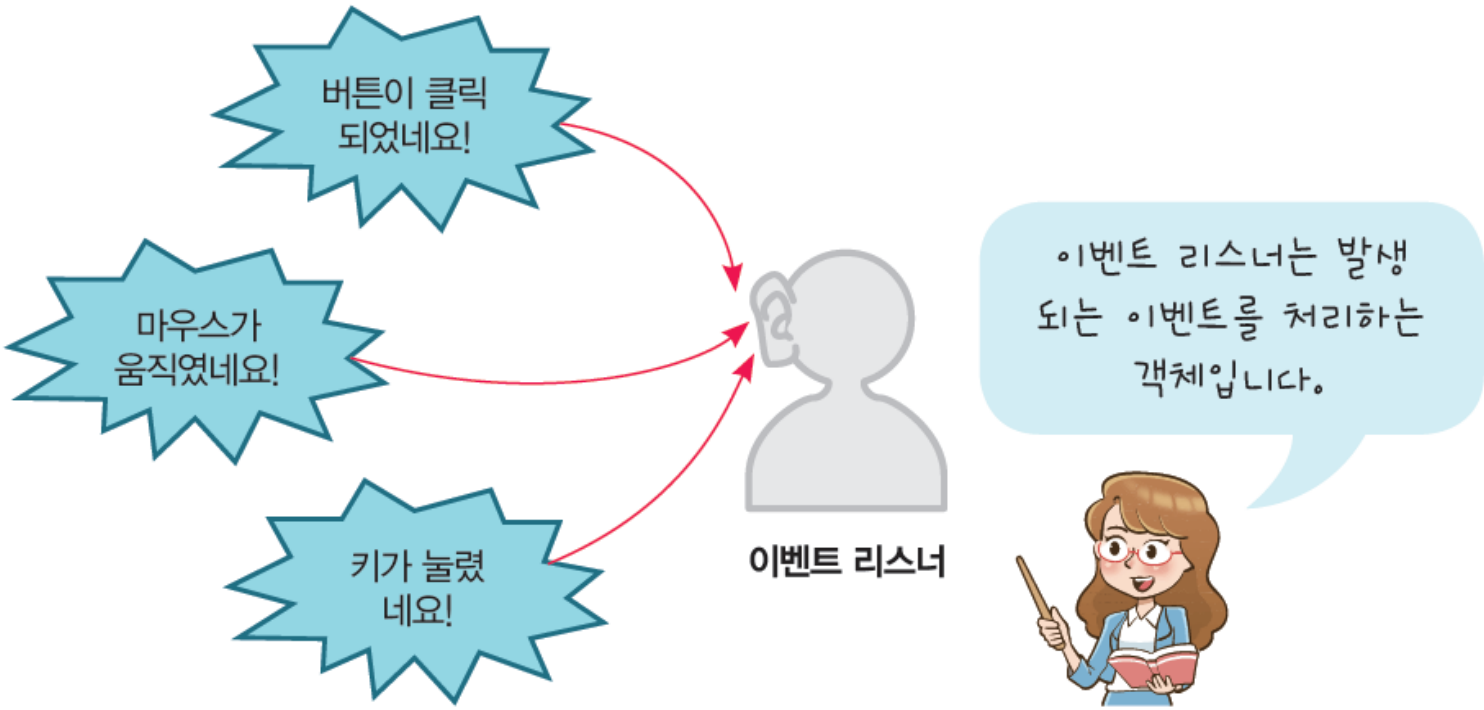
- Event-driven programming
- 프로그램의 실행이 이벤트의 발생에 의하여 결정되는 방식

# 이벤트 처리 과정


❶ 사용자가 버튼을 클릭한다.
Click!
❷ 이벤트가 발생한다.
이벤트
❸ 코드가 실행된다.
public void actionPerformed(ActionEvent e) {
...
}
❹ 어떤 작업을 실행한다.
이벤트 구동 프로그래밍은 이벤트에 의하여 실행 순서가 결정되는 방식입니다.

# 이벤트 리스너

- event listener
- 이벤트 발생 감지 후 발생한 이벤트를 처리하는 객체



# 이벤트 리스너 작성 과정

- 1. 이벤트 리스너 클래스 작성
  - 클래스를 이벤트 리스너로 만들기 위해 리스너 인터페이스 구현
  - ActionListener: 버튼 클릭과 같은 action event 처리 수행

```
class MyListener implements ActionListener {
  public void actionPerformed(ActionEvent e) {
    ... // Action 이벤트를 처리하는 코드가 여기에 들어간다.
  }
}
```

# 이벤트 리스너 작성 과정

- 2. 이벤트 리스너를 이벤트 소스에 등록
  - 이벤트 리스너 객체를 컴포넌트에 등록
  - 버튼 컴포넌트가 제공하는 addActionListener() 메소드를 통해 액션 이벤트 리스너 객체 등록

```
public class MyFrame extends JFrame {
  public MyFrame()  // 생성자에서 컴포넌트를 생성하고 추가한다.
  {
    button = new JButton("동작");  // 버튼 생성
    button.addActionListener(new MyListener());
            ...
  }
}
```

# 이벤트 처리 방법

**이벤트 처리 방법**

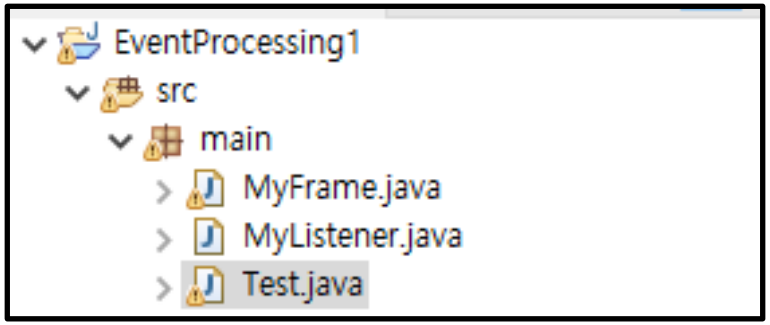
- (1) 독립적인 클래스로 이벤트 핸들러를 작성
- (2) 내부 클래스로 이벤트 핸들러를 작성
- (3) 프레임 클래스에 이벤트 처리를 구현
- (4) 무명 클래스를 사용하는 방법
- (5) 람다식을 이용하는 방법

# (1) 독립적인 클래스 작성

- Frame 이나 main() 메서드가 포함된 클래스와는 다른, 별도의 위치에 이벤트 리스너(혹은 이벤트 핸들러) 구현

# 예제: 독립적인 클래스 작성

- 프로젝트 구조



# 예제: 독립적인 클래스 작성

- MyListener.java

```
package main;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;

public class MyListener implements ActionListener{
  public void actionPerformed(ActionEvent e) {
    JButton button = (JButton) e.getSource();
    button.setText("You pressed the button.");
  }
}
```

# 예제: 독립적인 클래스 작성

- MyFrame.java

```java
package main;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

public class MyFrame extends JFrame{
  private JButton button;
  private JLabel label;

  public MyFrame() {
    setSize(300, 200);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setTitle("Event Processing Example");

    JPanel panel = new JPanel();
    button = new JButton("Press the button.");
    label = new JLabel("You did not press the button yet.");

    button.addActionListener(new MyListener());

    panel.add(button);
    panel.add(label);

    add(panel);
    setVisible(true);
  }
}
```

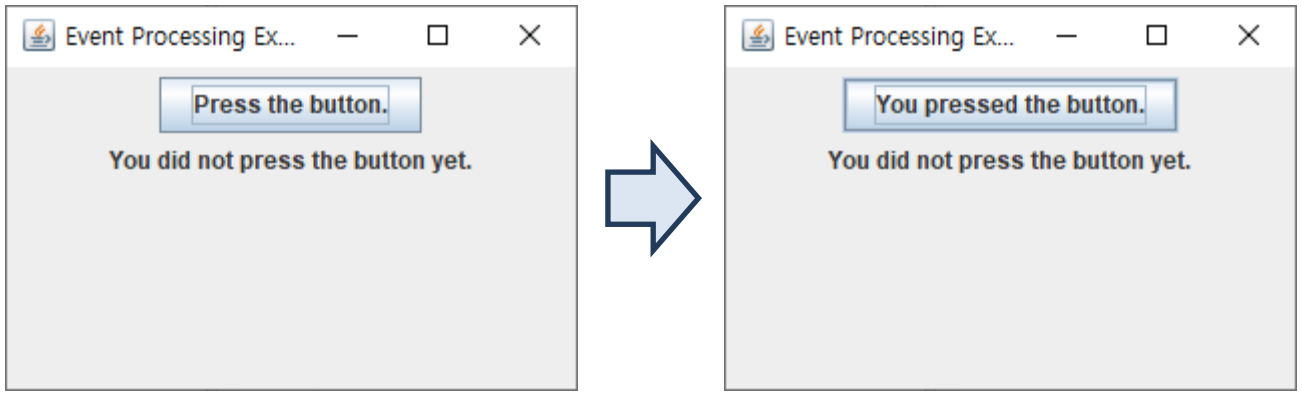
# 예제: 독립적인 클래스 작성

- Test.java

```
package main;

public class Test {
  public static void main(String[] args) {
    MyFrame f = new MyFrame();
  }
}
```

# 예제: 독립적인 클래스 작성

- Test.java 실행 결과

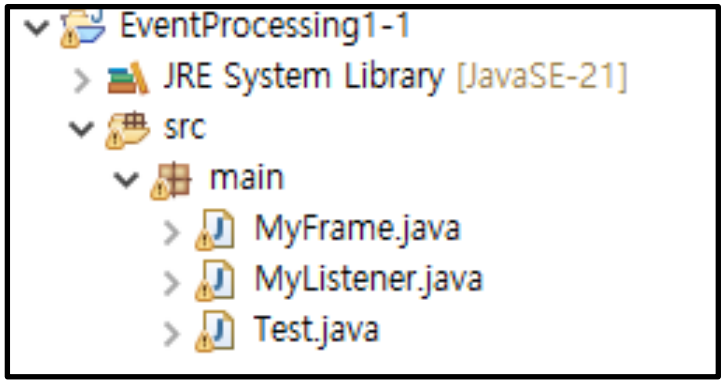


# (2) 내부 클래스 작성

- MyListener 클래스를 별도의 클래스로 만들면 MyFrame 안의 멤버 변수들을 쉽게 사용할 수 없음
  - MyListener 클래스 객체에 멤버 변수들을 전부 넘겨줘야 함
  - MyListener 클래스 객체가 MyFrame 객체에 의존적이게 됨
- 이 경우 MyListener 클래스를 내부 클래스로 만들어 문제 해결

# 예제: 내부 클래스를 사용하지 않는 경우

- 프로젝트 구조



# 예제: 내부 클래스를 사용하지 않는 경우

- MyListener.java

```
package main;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JLabel;

public class MyListener implements ActionListener{

  JLabel label;

  public MyListener(JLabel label) {
    this.label = label;
  }

  public void actionPerformed(ActionEvent e) {
    label.setText("You pressed the button.");
  }
}
```

# 예제: 내부 클래스를 사용하지 않는 경우

- MyFrame.java

```
package main;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

public class MyFrame extends JFrame{
  private JButton button;
  private JLabel label;

  public MyFrame() {
    setSize(300, 200);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setTitle("Event Processing Example");

    JPanel panel = new JPanel();
    button = new JButton("Press the button.");
    label = new JLabel("You did not press the button yet.");

    button.addActionListener(new MyListener(label));

    panel.add(button);
    panel.add(label);

    add(panel);
    setVisible(true);
  }
}
```

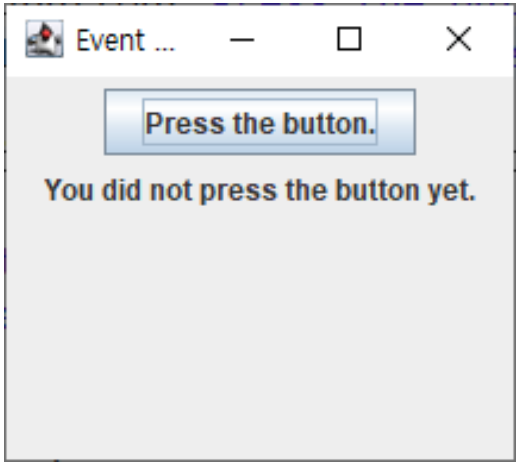
# 예제: 내부 클래스를 사용하지 않는 경우

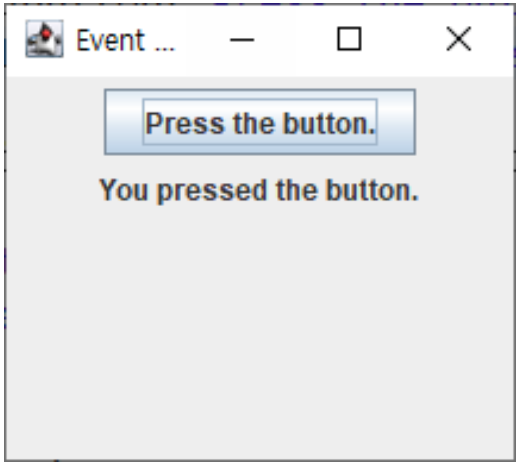
- Test.java

```
package main;

public class Test {
  public static void main(String[] args) {
    MyFrame f = new MyFrame();
  }
}
```

# 예제: 내부 클래스를 사용하지 않는 경우

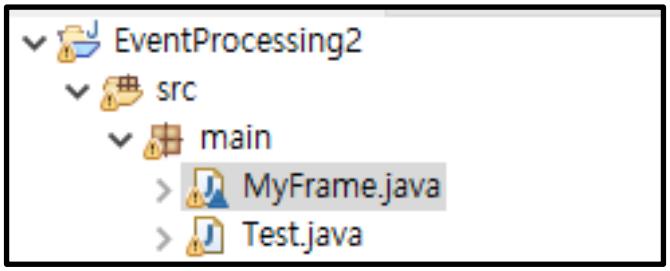
- Test.java 실행 결과



# 예제: 내부 클래스 작성

- 프로젝트 구조



# 예제: 내부 클래스 작성

- MyFrame.java (1/2)

```
package main;

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;

class MyFrame extends JFrame {
  private JButton button;
  private JLabel label;

  public MyFrame() {
    this.setSize(300, 200);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setTitle("Event Processing Example");

    JPanel panel = new JPanel();
    button = new JButton("Press the button.");
    label = new JLabel("You did not press the button yet.");

    button.addActionListener(new MyListener());
    panel.add(button);
    panel.add(label);
    this.add(panel);
    this.pack();
    this.setVisible(true);
  }
```

# 예제: 내부 클래스 작성

- MyFrame.java (2/2)

```
  private class MyListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
      if (e.getSource() == button) {
        label.setText("You pressed the button.");
      }
    }
  }
}
```

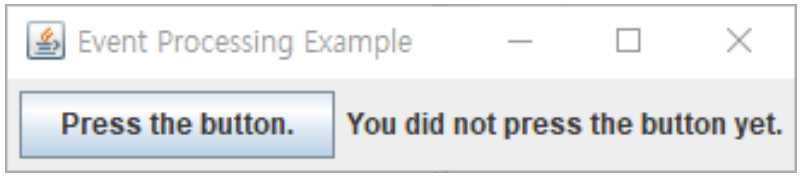
# 예제: 내부 클래스 작성

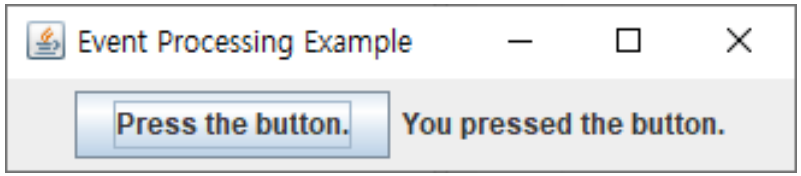
- Test.java

```
package main;

public class Test {
  public static void main(String[] args) {
    MyFrame t = new MyFrame();
  }
}
```

# 예제: 내부 클래스 작성

- Test.java 실행 결과



# (3) MyFrame에서 이벤트 처리

- MyFrame 클래스가 JFrame을 상속받으면서 동시에 ActionListener 인터페이스를 구현

```
...
class MyFrame extends JFrame implements ActionListener {
        ...
        public MyFrame() {
                ...
                button = new JButton("Press the button.");
                label = new JLabel("You did not press the button yet.");
                button.addActionListener(this);
                ...
        }
        public void actionPerformed(ActionEvent e) {
                if (e.getSource() == button) {
                        label.setText("You pressed the button.");
                }
        }
}
...
```

이벤트 처리

MyFrame이 핸들러 역할수행

# (4) 무명 클래스 사용

- ActionListener를 구현하는 무명 클래스 객체를 생성 후, addActionListener()의 파라미터로 전달

```
class MyFrame extends JFrame {
        ...
        public MyFrame() {
                ...
                button = new JButton("Press the button.");
                button.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                if (e.getSource() == button) {
                                        label.setText("You pressed the button.");
                                }
                        }
                });
                ...
        }
}
```

**안드로이드에서 많이 사용된다!**

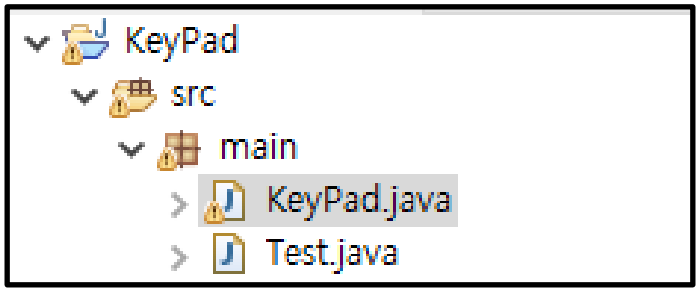
# (5) 람다식을 이용

- 함수 인터페이스인 경우(예:ActionListener) 람다식을 이용하여 무명 이벤트 리스너를 생성 및 등록할 수 있음

```
import javax.swing.*;
class MyFrame extends JFrame {
		private JButton button;
		private JLabel label;
		public MyFrame() {
				this.setSize(300, 200);
				this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
				this.setTitle("Event Processing Example");
				JPanel panel = new JPanel();
				button = new JButton("Press the button.");
				label = new JLabel("You did not press the button.");
				button.addActionListener(e -> {
						label.setText("You pressed the button.");
				});
				panel.add(button);
				panel.add(label);
				this.add(panel);
				this.setVisible(true);
		}
}
```

# 예제: 키패드 만들기

- 프로젝트 구조



# 예제: 키패드 만들기

- KeyPad.java (1/2)

```
package main;

import java.awt.BorderLayout;
import java.awt.Dimension;
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.JPanel;
import javax.swing.JTextField;
```

# 예제: 키패드 만들기

- KeyPad.java (2/2)

```
public class KeyPad extends JFrame implements ActionListener {
  private JTextField txt;
  private JPanel panel;

  public KeyPad() {
    txt = new JTextField(20);
    add(txt, BorderLayout.NORTH);
    panel = new JPanel();
    panel.setLayout(new GridLayout(3, 3));
    add(panel, BorderLayout.CENTER);
    for (int i = 1; i <= 9; i++) {
      JButton btn = new JButton("" + i);
      btn.addActionListener(this);
      btn.setPreferredSize(new Dimension(100, 100));
      panel.add(btn);
    }
    pack();
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setVisible(true);
  }

  @Override
  public void actionPerformed(ActionEvent e) {
    String actionCommand = e.getActionCommand();
    txt.setText(txt.getText() + actionCommand);
  }
}
```

# 예제: 키패드 만들기

- Test.java

```
package main;

public class Test {
  public static void main(String[] args) {
    new KeyPad();
  }
}
```

# 예제: 키패드 만들기

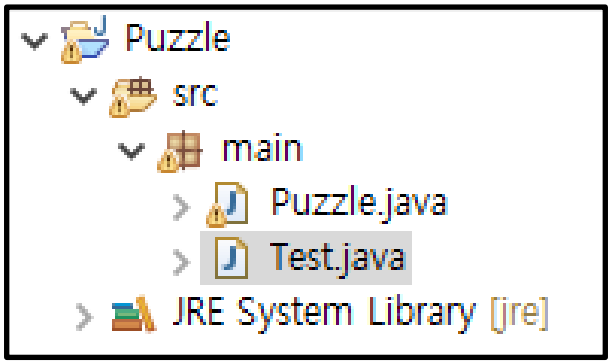
- Test.java 실행 결과

# 예제: 퍼즐게임

- 타일을 클릭하면 비어있는 옆의 공간으로 이동하는 퍼즐 게임 작성
  - sliding puzzle 이라고 함

# 예제: 퍼즐게임

- 프로젝트 구조



# 예제: 퍼즐게임

- Puzzle.java (1/6)

```
package main;

import java.awt.BorderLayout;
import java.awt.Color;
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.JPanel;
```

# 예제: 퍼즐게임

- Puzzle.java (2/6)

```
public class Puzzle extends JFrame implements ActionListener {
    MyButton[] buttons;
    MyButton reset;

    public Puzzle() {
        super("puzzle");
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(0, 3, 2, 2));
        buttons = new MyButton[9];

        for (int i = 0; i < 8; i++)
            buttons[i] = new MyButton("" + (i + 1));
        buttons[8] = new MyButton(" ");
        for (int i = 0; i < 9; i++)
            panel.add(buttons[i]);
        for (int i = 0; i < 9; i++)
            buttons[i].addActionListener(this);

        add(panel, BorderLayout.CENTER);
        reset = new MyButton("reset");
        reset.setBackground(Color.red);
        reset.setForeground(Color.yellow);
        add(reset, BorderLayout.SOUTH);

        setSize(300, 300);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```

# 예제: 퍼즐게임

- Puzzle.java (3/6)

```
public void actionPerformed(ActionEvent e) {
  MyButton b = (MyButton) e.getSource();
  if( b.getText().equals(" ")==true) return;
  if( b.index == 0 ){
    if( buttons[1].getText().equals(" ") )
    { buttons[1].setText(b.getText()); b.setText(" "); }
    if( buttons[3].getText().equals(" ") )
    { buttons[3].setText(b.getText()); b.setText(" "); }
  }
  if( b.index == 1 ){
    if( buttons[0].getText().equals(" ") )
    { buttons[0].setText(b.getText()); b.setText(" "); }
    if( buttons[2].getText().equals(" ") )
    { buttons[2].setText(b.getText()); b.setText(" "); }
    if( buttons[4].getText().equals(" ") )
    { buttons[4].setText(b.getText()); b.setText(" "); }
  }
  if( b.index == 2 ){
    if( buttons[1].getText().equals(" ") )
    { buttons[1].setText(b.getText()); b.setText(" "); }
    if( buttons[5].getText().equals(" ") )
    { buttons[5].setText(b.getText()); b.setText(" "); }
  }
```

# 예제: 퍼즐게임

- Puzzle.java (4/6)

```
if( b.index == 3 ){
  if( buttons[0].getText().equals(" ") )
  { buttons[0].setText(b.getText()); b.setText(" "); }
  if( buttons[4].getText().equals(" ") )
  { buttons[4].setText(b.getText()); b.setText(" "); }
  if( buttons[6].getText().equals(" ") )
  { buttons[6].setText(b.getText()); b.setText(" "); }
}
if( b.index == 4 ){
  if( buttons[1].getText().equals(" ") )
  { buttons[1].setText(b.getText()); b.setText(" "); }
  if( buttons[3].getText().equals(" ") )
  { buttons[3].setText(b.getText()); b.setText(" "); }
  if( buttons[5].getText().equals(" ") )
  { buttons[5].setText(b.getText()); b.setText(" "); }
  if( buttons[7].getText().equals(" ") )
  { buttons[7].setText(b.getText()); b.setText(" "); }
}
if( b.index == 5 ){
  if( buttons[2].getText().equals(" ") )
  { buttons[2].setText(b.getText()); b.setText(" "); }
  if( buttons[4].getText().equals(" ") )
  { buttons[4].setText(b.getText()); b.setText(" "); }
  if( buttons[8].getText().equals(" ") )
  { buttons[8].setText(b.getText()); b.setText(" "); }
}
```

# 예제: 퍼즐게임

- Puzzle.java (5/6)

```
  if( b.index == 6 ){
    if( buttons[3].getText().equals(" ") )
    { buttons[3].setText(b.getText()); b.setText(" "); }
    if( buttons[7].getText().equals(" ") )
    { buttons[7].setText(b.getText()); b.setText(" "); }
  }
  if( b.index == 7 ){
    if( buttons[4].getText().equals(" ") )
    { buttons[4].setText(b.getText()); b.setText(" "); }
    if( buttons[6].getText().equals(" ") )
    { buttons[6].setText(b.getText()); b.setText(" "); }
    if( buttons[8].getText().equals(" ") )
    { buttons[8].setText(b.getText()); b.setText(" "); }
  }
  if( b.index == 8 ){
    if( buttons[5].getText().equals(" ") )
    { buttons[5].setText(b.getText()); b.setText(" "); }
    if( buttons[7].getText().equals(" ") )
    { buttons[7].setText(b.getText()); b.setText(" "); }
  }
}
```

# 예제: 퍼즐게임

- Puzzle.java (6/6)

```
class MyButton extends JButton {
    static int count = 0;
    int index;

    public MyButton(String s) {
        super(s);
        index = count++;
    }
}
```

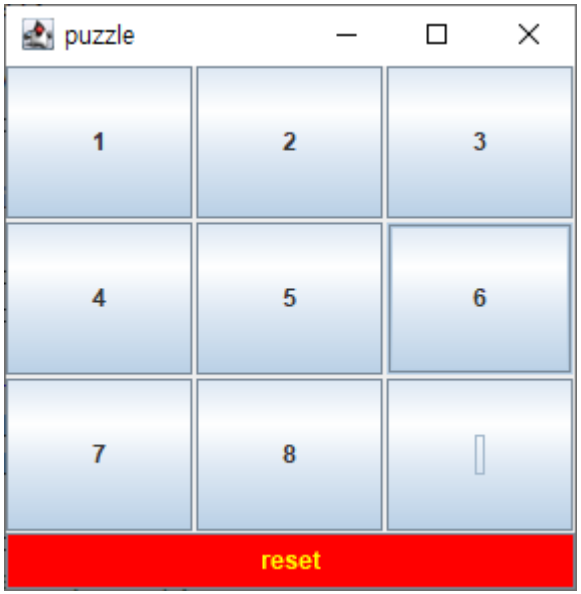
# 예제: 퍼즐게임

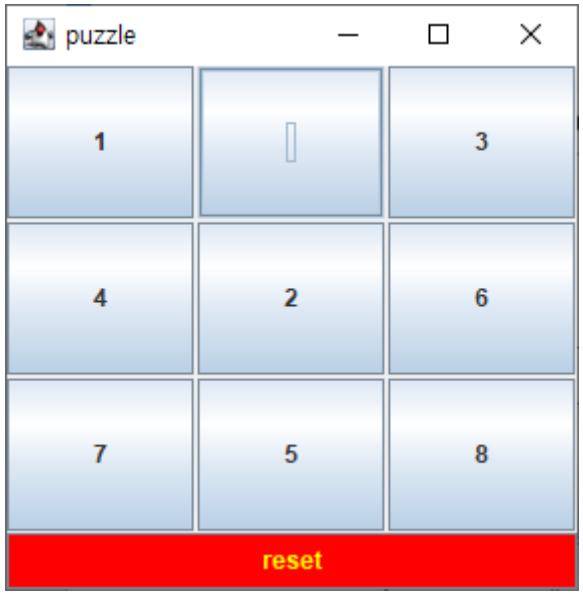
- Test.java

```
package main;

public class Test {
    public static void main(String[] args) {
        new Puzzle();
    }
}
```

# 예제: 퍼즐게임

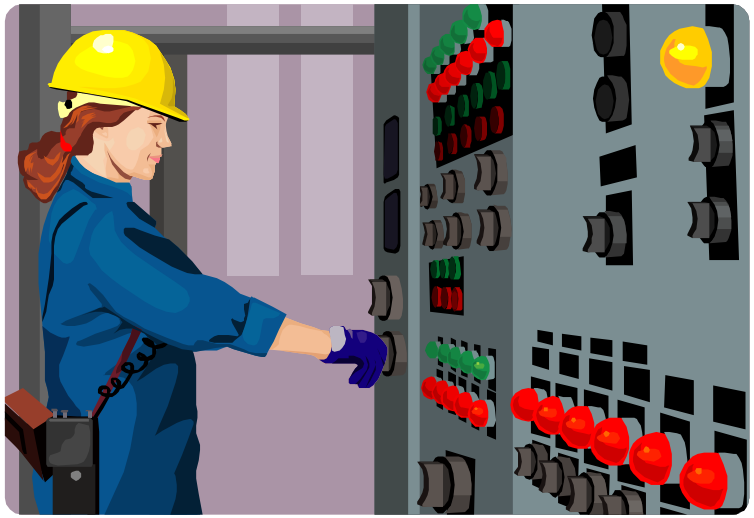
- Test.java 실행 결과
  - reset은 눌러도 반응없음



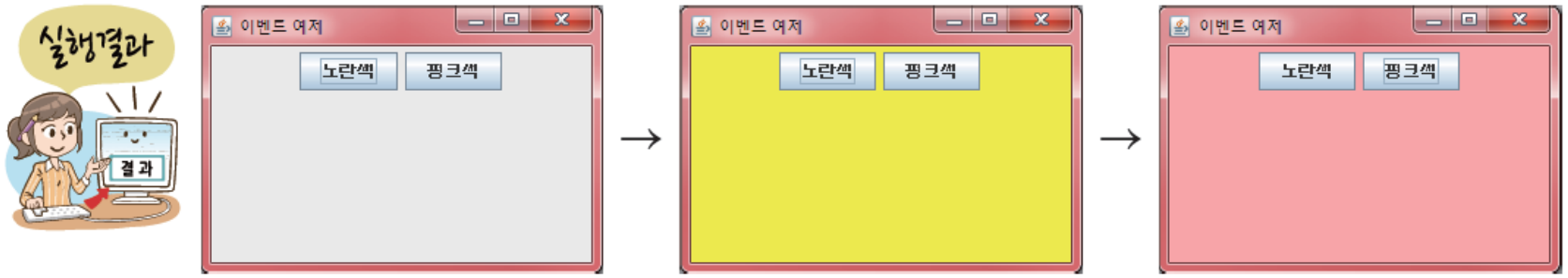
# 액션 이벤트

- action event
  - 사용자가 버튼을 클릭하는 경우
  - 사용자가 메뉴 항목을 선택하는 경우
  - 사용자가 텍스트 필드에서 엔터키를 누르는 경우



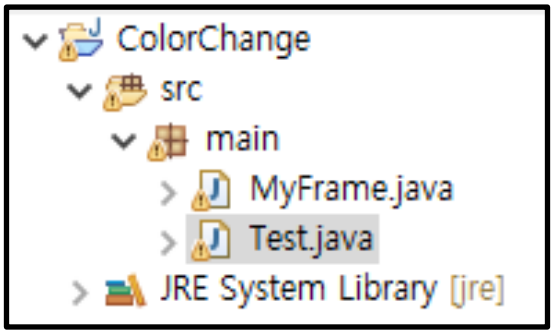
# 예제: 액션 이벤트

- 두 개의 버튼을 만들어서 패널의 배경 색을 변경하는 프로그램 작성
- 이벤트 리스너는 하나만 생성



# 예제: 액션 이벤트

- 프로젝트 구조



# 예제: 액션 이벤트

- MyFrame.java (1/2)

```
package main;

import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
```

# 예제: 액션 이벤트

- MyFrame.java (2/2)

```
public class MyFrame extends JFrame {
    private JButton button1;
    private JButton button2;
    private JPanel panel;

    public MyFrame() {
        this.setSize(300, 200);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setTitle("Event Processing Example");
        panel = new JPanel();
        button1 = new JButton("Yellow");
        button1.addActionListener(new MyListener());
        panel.add(button1);
        button2 = new JButton("Pink");
        button2.addActionListener(new MyListener());
        panel.add(button2);
        this.add(panel);
        this.setVisible(true);
    }

    private class MyListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            if (e.getSource() == button1) {
                panel.setBackground(Color.YELLOW);
            } else if (e.getSource() == button2) {
                panel.setBackground(Color.PINK);
            }
        }
    }
}
```

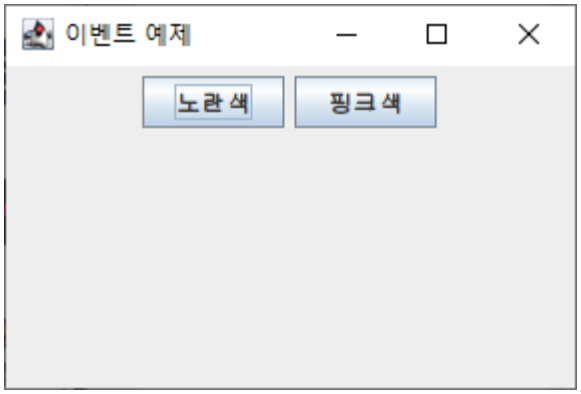
# 예제: 액션 이벤트

- Test.java

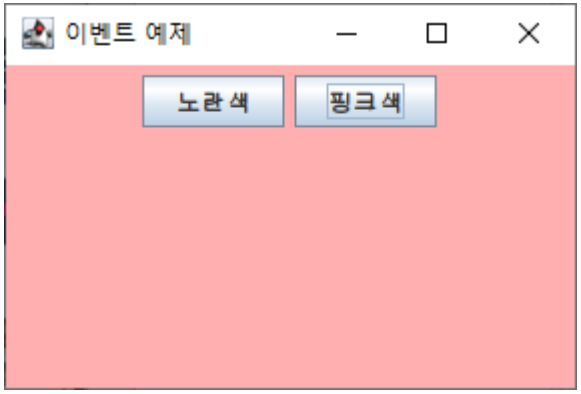
```
package main;

public class Test {
    public static void main(String[] args) {
        MyFrame t = new MyFrame();
  }
}
```

# 예제: 액션 이벤트

- Test.java 실행 결과







# 이벤트 발생 소스 식별

- getSource(): 이벤트를 발생시킨 객체 식별
- getId(): 이벤트의 타입 식별
- getActionCommand(): 이벤트를 발생시킨 컴포넌트 이름 식별

```
public void actionPerformed(ActionEvent e) {
        if (e.getSource () == button1){
                ...
        }
}
```

# 키 이벤트

- KeyListener 인터페이스 구현

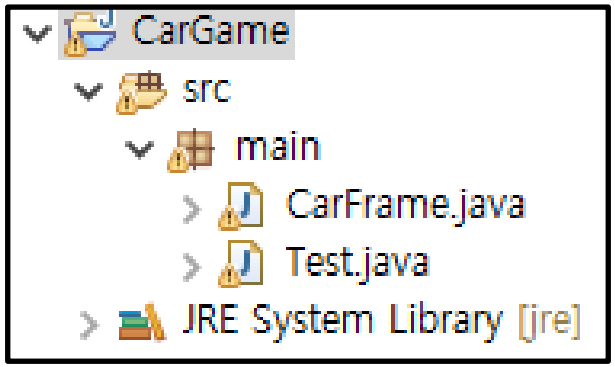
| 메소드 | 설 명 |
|---|---|
| keyTyped(KeyEvent e) | 사용자가 글자를 입력했을 경우에 호출 |
| keyPressed(KeyEvent e) | 사용자가 키를 눌렀을 경우에 호출 |
| keyReleased(KeyEvent e) | 사용자가 키에서 손을 떼었을 경우에 호출 |

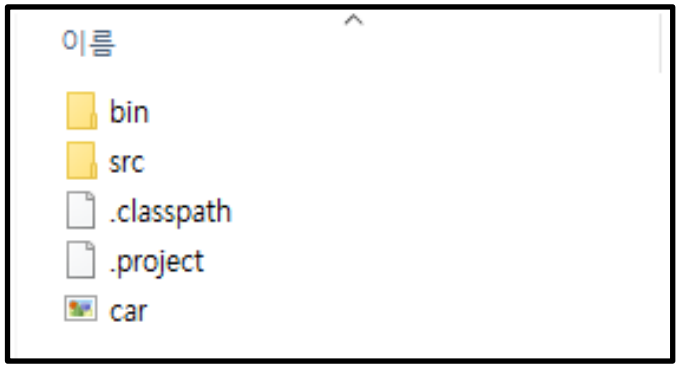
# 예제: 자동차 게임

- 키보드의 화살표 키로 움직이는 자동차 애플리케이션 작성

# 예제: 자동차 게임

- 프로젝트 구조




car.gif 파일

# 예제: 자동차 게임

- CarFrame.java (1/2)

```
package main;

import java.awt.Graphics;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;
import javax.swing.JFrame;
import javax.swing.JPanel;

public class CarFrame extends JFrame {
    public CarFrame() {
        setSize(300, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        add(new CarPanel());
        setVisible(true);
    }
}
```

# 예제: 자동차 게임

- CarFrame.java (2/2)

```
class CarPanel extends JPanel {
    BufferedImage img = null;
    int img_x = 100, img_y = 100;

    public CarPanel() {
        try {
            img = ImageIO.read(new File("car.gif"));
        } catch (IOException e) {
            System.out.println("no image");
            System.exit(1);
        }
        addKeyListener(new KeyListener() {
            public void keyPressed(KeyEvent e) {
                int keycode = e.getKeyCode();
                switch (keycode) {
                case KeyEvent.VK_UP:
                    img_y -= 10; break;
                case KeyEvent.VK_DOWN:
                    img_y += 10; break;
                case KeyEvent.VK_LEFT:
                    img_x -= 10; break;
                case KeyEvent.VK_RIGHT:
                    img_x += 10; break;
                }
                repaint();
            }
            public void keyReleased(KeyEvent arg0) {}
            public void keyTyped(KeyEvent arg0) {}
        });
        this.requestFocus();
        setFocusable(true);
    }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawImage(img, img_x, img_y, null);
    }
}
```

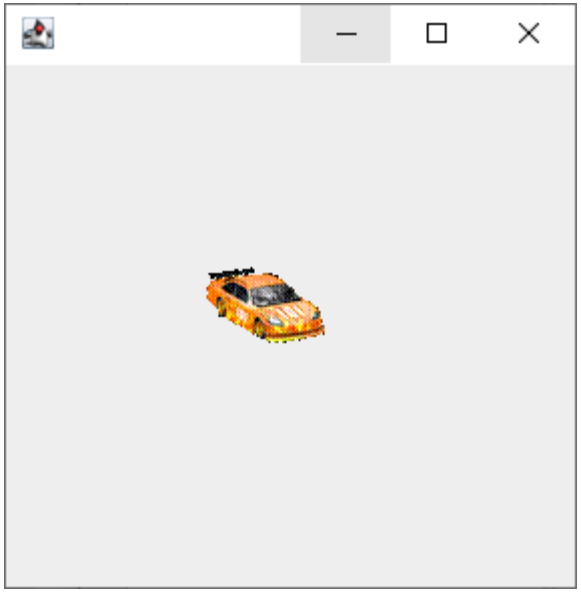
# 예제: 자동차 게임

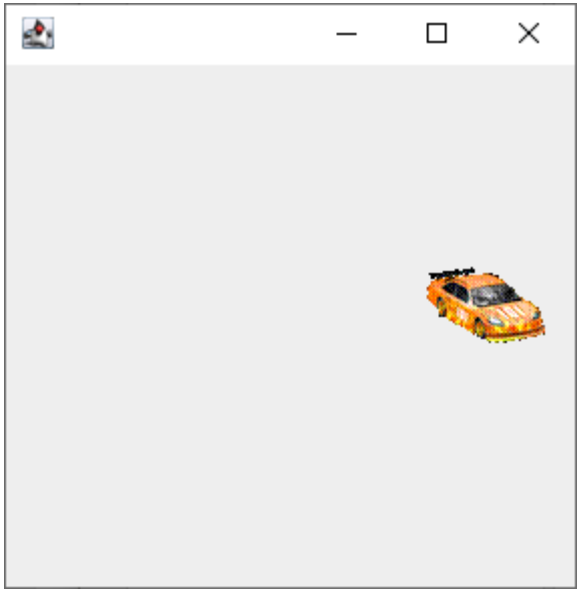
- Test.java

```
package main;

public class Test {
    public static void main(String[] args) {
        CarFrame f = new CarFrame();
    }
}
```

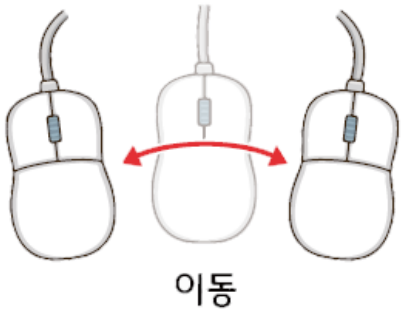
# 예제: 자동차 게임

- Test.java 실행 결과





# 마우스 이벤트





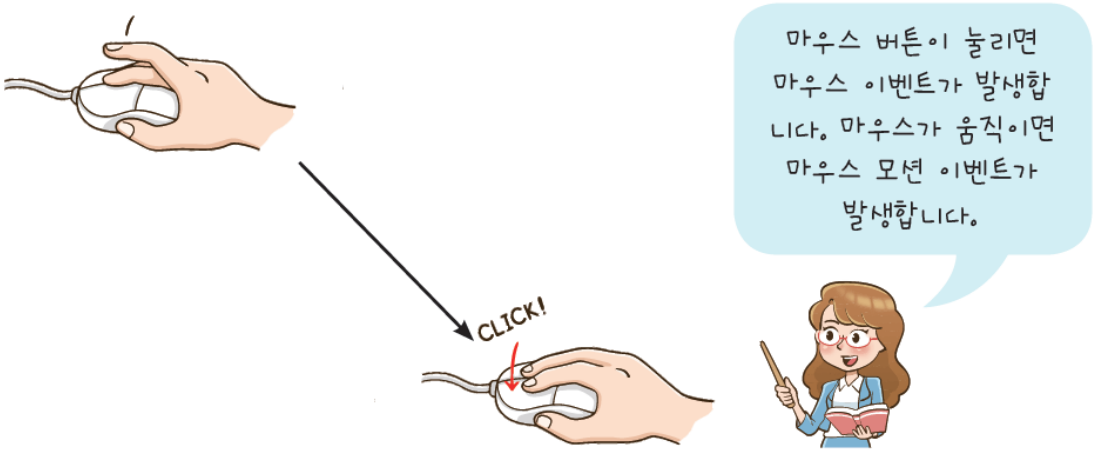


**마우스 클릭 이벤트 발생 순서**

1. Mouse pressed
2. Mouse released
3. Mouse clicked

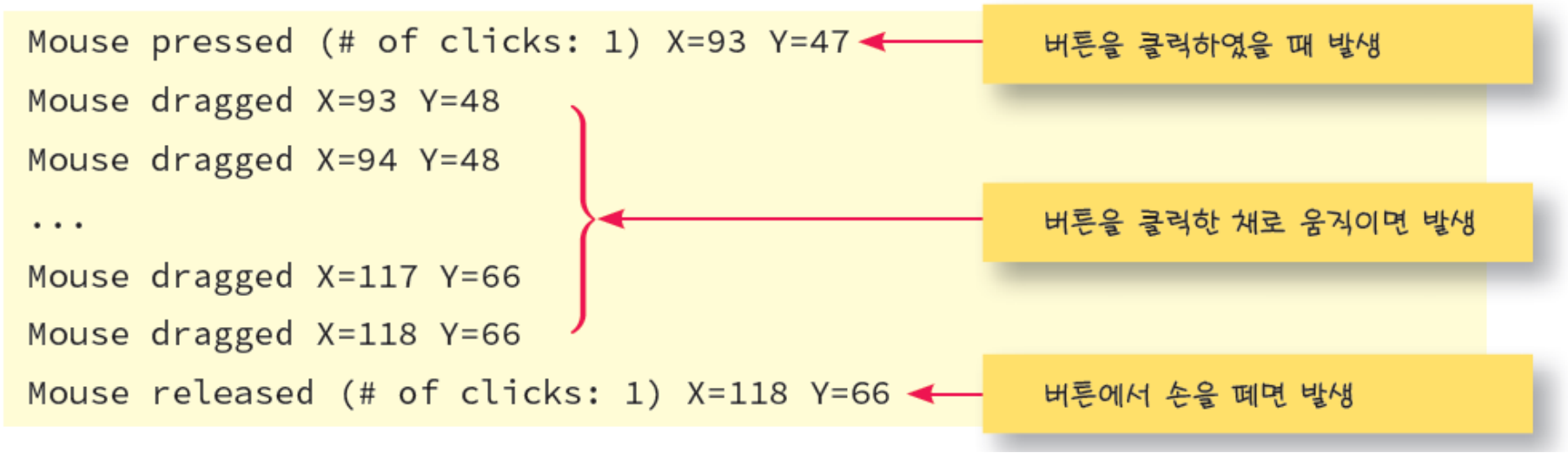
# MouseListener 인터페이스

| 메소드 | 설 명 |
|---|---|
| mouseClicked(MouseEvent e) | 사용자가 컴포넌트를 클릭한 경우에 호출된다. |
| mouseEntered(MouseEvent e) | 마우스 커서가 컴포넌트로 들어가면 호출된다. |
| mouseExited(MouseEvent e) | 마우스 커서가 컴포넌트에서 나가면 호출된다. |
| mousePressed(MouseEvent e) | 마우스가 컴포넌트위에서 눌려지면 호출된다. |
| mouseReleased(MouseEvent e) | 마우스가 컴포넌트위에서 떼어지면 호출된다. |



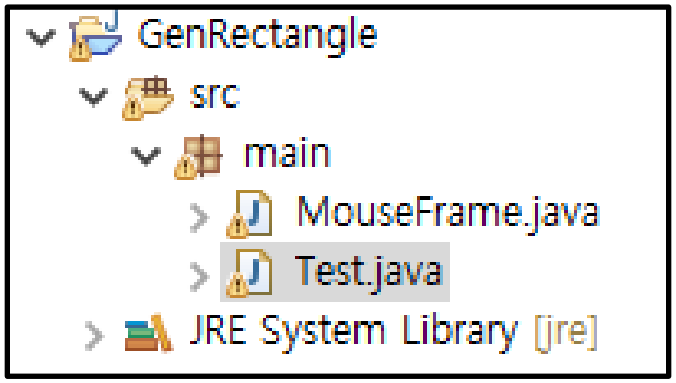
# MouseMotionListener 인터페이스

| 메소드 | 설 명 |
|---|---|
| mouseDragged(MouseEvent e) | 마우스 드래그하면 호출된다. |
| mouseMoved(MouseEvent e) | 마우스가 클릭되지 않고 이동하는 경우에 호출된다. |



# 예제: 사각형 그리기

- 프로젝트 구조



# 예제: 사각형 그리기

- MouseFrame.java (1/3)

```
package main;

import java.awt.Graphics;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.image.BufferedImage;

import javax.swing.JFrame;
import javax.swing.JPanel;

public class MouseFrame extends JFrame {
    public MouseFrame() {
        setSize(300, 300);
        setTitle("Draw the rectangle");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        add(new MyPanel());
        setVisible(true);
    }
}
```

# 예제: 사각형 그리기

- MouseFrame.java (2/3)

```
class MyPanel extends JPanel implements MouseListener {
    BufferedImage img = null;
    int img_x = 0, img_y = 0;

    Rectangle[] array = new Rectangle[100];
    int index = 0;

    public MyPanel() { this.addMouseListener(this); }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        for (Rectangle r : array)
            if (r != null) g.drawRect(r.x, r.y, r.w, r.h);
    }
```

# 예제: 사각형 그리기

- MouseFrame.java (3/3)

```
    @Override
    public void mousePressed(MouseEvent e) {
        if (index > 100) return;
        array[index] = new Rectangle();
        array[index].x = e.getX(); array[index].y = e.getY();
        array[index].w = 50; array[index].h = 50;
        index++;
        repaint();
    }
    @Override
    public void mouseReleased(MouseEvent e) {}
    @Override
    public void mouseClicked(MouseEvent e) {}
    @Override
    public void mouseEntered(MouseEvent e) {}
    @Override
    public void mouseExited(MouseEvent e) {}
}

class Rectangle {
    int x, y, w, h;
}
```

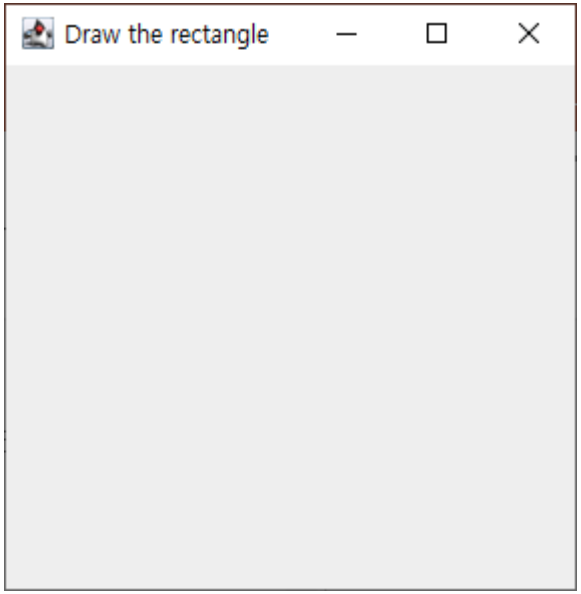
# 예제: 사각형 그리기

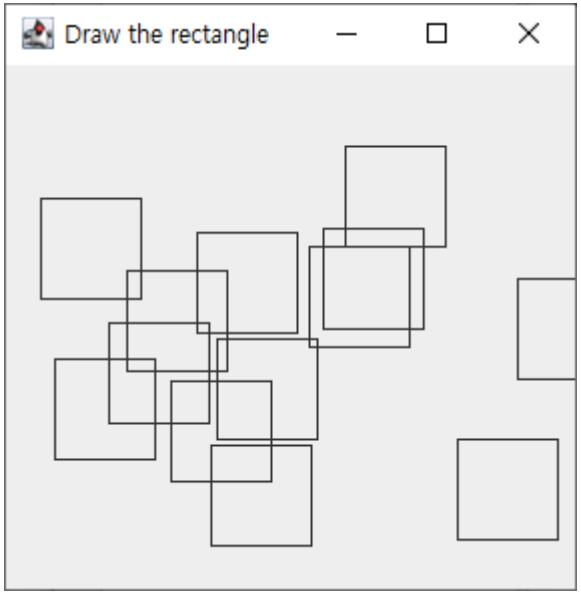
- Test.java

```
package main;

public class Test {
    public static void main(String[] args) {
        MouseFrame f = new MouseFrame();
    }
}
```

# 예제: 사각형 그리기

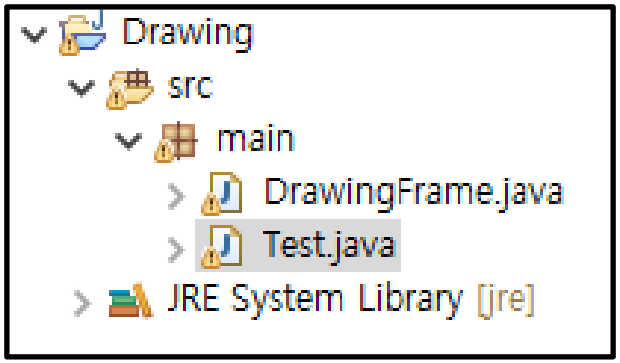
- Test.java 실행 결과



# 예제: 그림 그리기

- 프로젝트 구조



# 예제: 그림 그리기

- DrawingFrame.java (1/2)

```
package main;

import java.awt.Graphics;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionListener;

import javax.swing.JFrame;
import javax.swing.JPanel;

public class DrawingFrame extends JFrame {
    public DrawingFrame() {
        setSize(300, 300);
        setTitle("Drawing");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        add(new MyPanel());
        setVisible(true);
    }
}
```

# 예제: 그림 그리기

- DrawingFrame.java (2/2)

```
class MyPanel extends JPanel implements MouseMotionListener {
    private int index = 0;
    Point[] array = new Point[1000];

    public MyPanel() {
        this.addMouseMotionListener(this);
    }

    @Override
    public void mouseDragged(MouseEvent e) {
        int x = e.getX();
        int y = e.getY();
        if (index > 1000) return;
        array[index] = new Point();
        array[index].x = e.getX();
        array[index].y = e.getY();
        index++;
        repaint();
    }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        for (Point p : array)
            if (p != null) g.drawRect(p.x, p.y, 1, 1);
    }

    @Override
    public void mouseMoved(MouseEvent arg0) {}
}

class Point {
    int x, y;
}
```

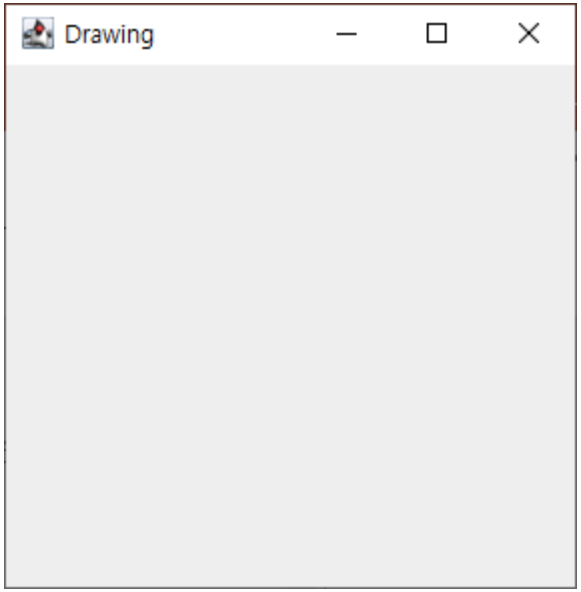
# 예제: 그림 그리기

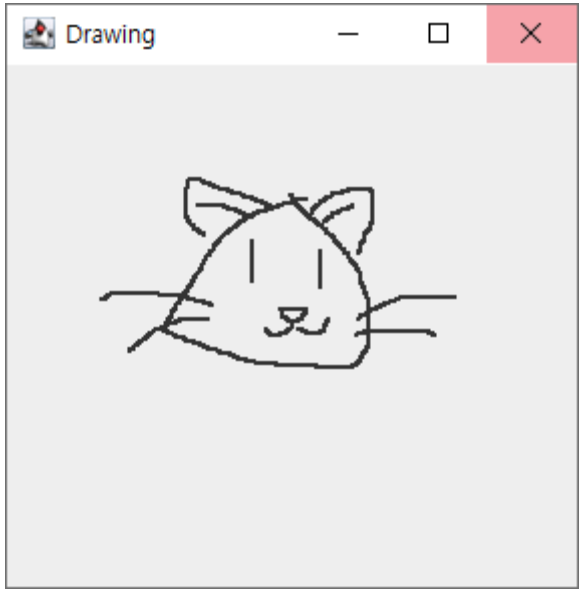
- Test.java

```
package main;

public class Test {
    public static void main(String[] args) {
        DrawingFrame f = new DrawingFrame();
    }
}
```

# 예제: 그림 그리기

- Test.java 실행 결과



# 어댑터 클래스

- 인터페이스의 경우, 모든 메소드를 구현하여야 함
- 어댑터 클래스(Adaptor Class)를 사용하면 원하는 메소드 만을 구현 가능

| 인터페이스 | 어댑터 클래스 |
|---|---|
| ComponentListener | ComponentAdapter |
| ContainerListener | ContainerAdapter |
| FocusListener | FocusAdater |
| KeyListener | KeyAdapter |
| MouseListener | MouseAdapter |
| MouseMotionListener | MouseMotionAdapter |
| WindowListener | WindowAdapter |

# 리스너를 사용하는 경우

```
public class MyClass implements MouseListener {
        public MyClass() {
                // ...
                someObject.addMouseListener(this);
        }
        public void mousePressed(MouseEvent e) {    }
        public void mouseReleased(MouseEvent e) {   }
        public void mouseEntered(MouseEvent e) {    }
        public void mouseExited(MouseEvent e) { }
        public void mouseClicked(MouseEvent e) {
                // ...
                // ...
        }
}
```

# 어댑터를 사용하는 경우

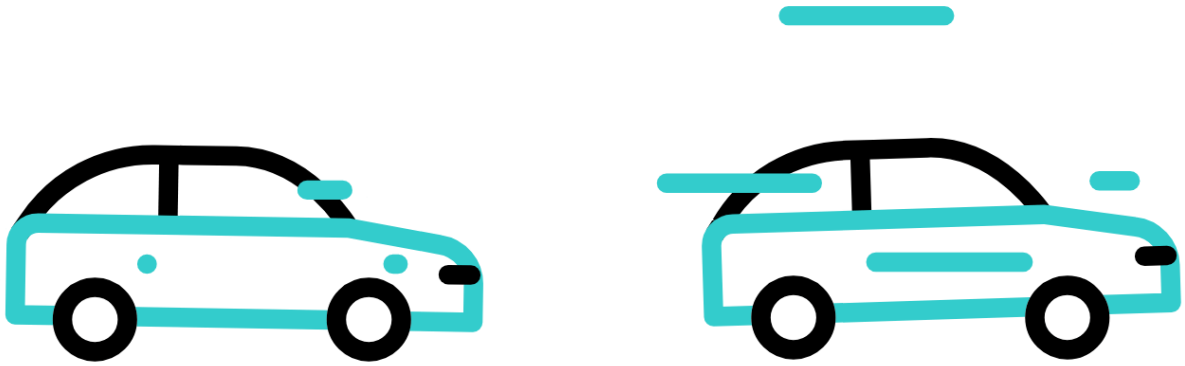
```
public class MyClass extends MouseAdapter {
        public MyClass() {
                // ...
                someObject.addMouseListener(this);
        }
        public void mouseClicked(MouseEvent e) {
                // ...
        }
}
```

# 문제 1

- p28 - 37의 Puzzle.java 파일을 수정해보자
  - 퍼즐판의 버튼 폰트(크기, 색깔) 및 배경색을 바꾸어보자.
  - reset 버튼 이벤트 처리 코드를 구현해서 클릭시 초기화면으로 돌아가도록 하자.

# 문제 2

- p47 - 52의 CarFrame.java 파일을 수정해보자
  - 자동차 이미지를 바꾸어서 표현해보자.
  - 자동차 바퀴가 굴러가는 효과를 주려면 어떻게 해야할까?
  - 키보드 입력에 따라 다양한 움직임이 가능하도록 해보자 (가속, 점프 등)



# 감사합니다! XD