Chapter 15

이벤트 처리

Contents

이벤트의 개념과 처리 과정

- 윈도우 시스템에서 사용자의 움직임을 애플리케이션에 전달하는 일종의 신호
- GUI프로그램은 이벤트가 실행 흐름을 결정하는 이벤트 구동 방식
- 이벤트 구동 프로그램의 이벤트 처리 과정

- 이벤트 리스너는 발생한 이벤트를 처리하는 객체
- 이벤트 핸들러는 이벤트를 처리하는 이벤트 리스너의 멤버 메서드

이벤트 맛보기(1)

- 예제 : [예제 15-1]

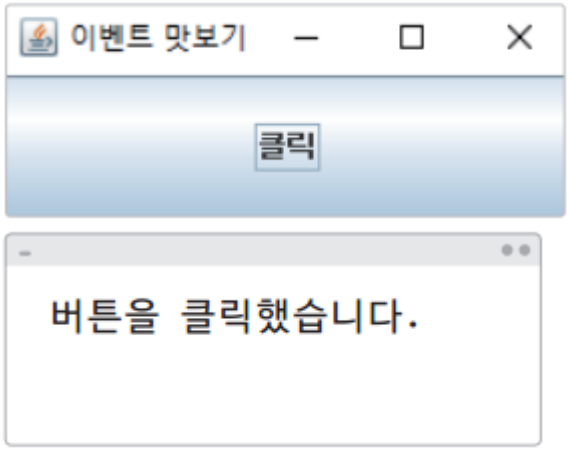
```
07  public class HelloEventDemo extends JFrame {
08    HelloEventDemo() {
09       setTitle("이벤트 맛보기");
10
11       ActionListener l = new ActionListener() {
12          public void actionPerformed(ActionEvent e) {
13             System.out.println("버튼을 클릭했습니다.");
14          }
15       };
16
17       JButton b = new JButton("클릭");
18       b.addActionListener(l);
19
20       add(b);
21
```

이벤트 리스너 구현 객체를 생성한다.

이벤트 소스로 사용할 버튼이다. 버튼을 클릭하면 ActionEvent를 발생한다.

이벤트 소스인 버튼에 이벤트 리스너를 등록한다.

```
22        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
23        setSize(260, 100);
24        setVisible(true);
25     }
26
27     public static void main(String[] args) {
28        new HelloEventDemo();
29     }
30  }
```



이벤트 맛보기(2)

```
11      ActionListener l = new ActionListener() {
12         public void actionPerformed(ActionEvent e) {
13            System.out.println("버튼을 클릭했습니다.");
14         }
15      };
16
17      JButton b = new JButton("클릭");
18      b.addActionListener(l);
```

이벤트 리스너 구현 객체를 생성한다.

이벤트 소스로 사용할 버튼이다. 버튼을 클릭하면 ActionEvent를 발생한다.

이벤트 소스인 버튼에 이벤트 리스너를 등록한다.

```
ActionListener l = e -> System.out.println("버튼을 클릭했습니다.");
```

```
JButton b = new JButton("클릭");
b.addActionListener(e -> System.out.println("버튼을 클릭했습니다."));
```

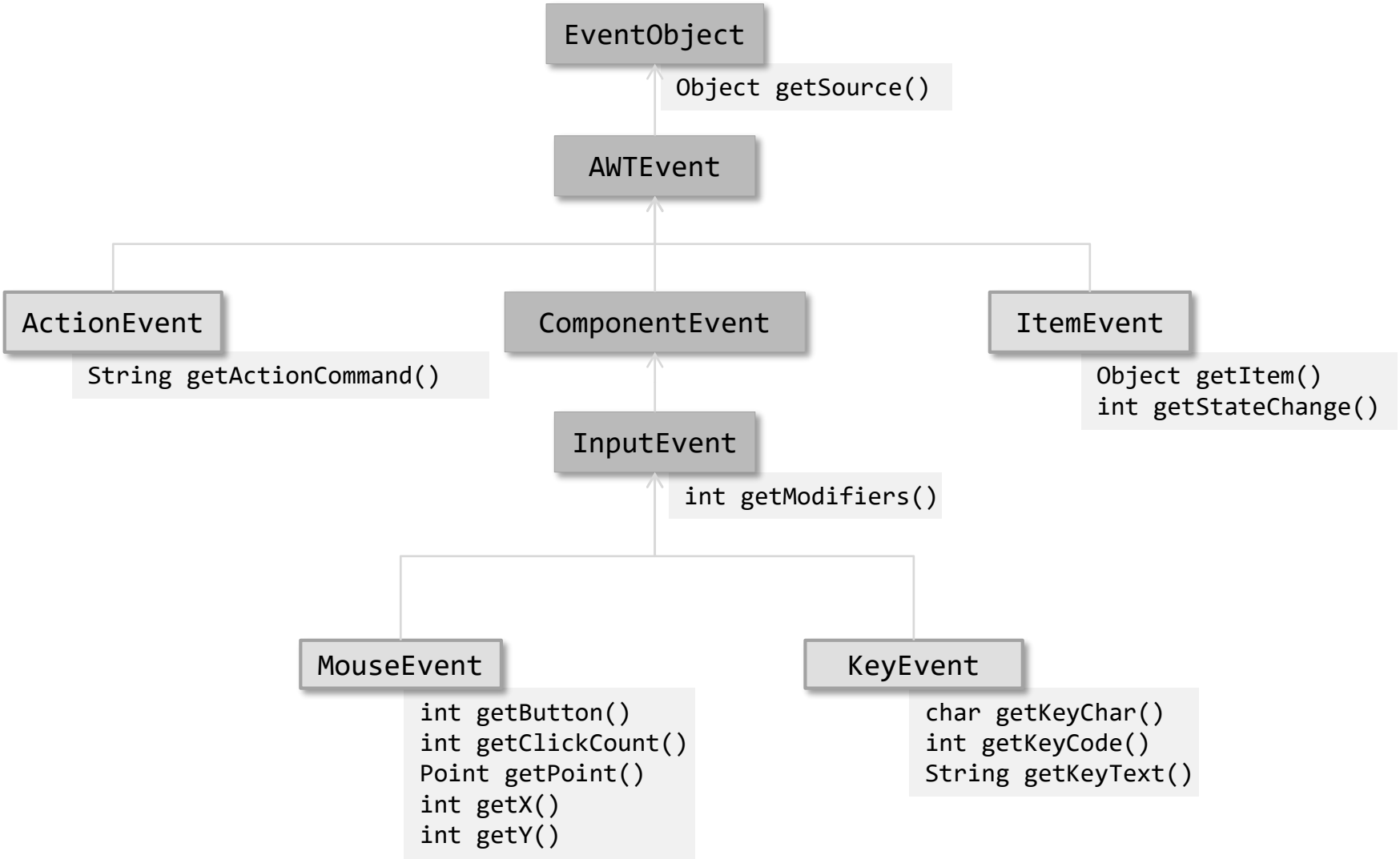
이벤트 객체

- 이벤트 객체
  - 발생한 이벤트에 관한 정보를 가진 객체
  - 이벤트 리스너에 전달됨
    - 이벤트 리스너 코드가 발생한 이벤트에 대한 상황을 파악할 수 있게 함

- 이벤트 객체가 포함하는 정보
  - 이벤트 종류와 이벤트 소스
  - 이벤트가 발생한 화면 좌표 및 컴포넌트 내 좌표
  - 이벤트가 발생한 버튼이나 메뉴 아이템의 문자열
  - 클릭된 마우스 버튼 번호 및 마우스의 클릭 횟수
  - 키의 코드 값과 문자 값
  - 체크박스, 라디오버튼 등과 같은 컴포넌트에 이벤트가 발생하였다면 체크 상태

- 이벤트 소스를 알아 내는 메소드
  - `Object getSource()`
    - 발생한 이벤트의 소스 컴포넌트 리턴
    - `Object` 타입으로 리턴하므로 캐스팅하여 사용
    - 모든 이벤트 객체에 대해 적용

이벤트 클래스와 이벤트 정보를 리턴하는 메소드



이벤트 클래스의 종류

- 의미 이벤트
  - 버튼 클릭처럼 사용자가 의도하는 이벤트를 의미
  - 일부 스윙 컴포넌트에서만 발생
  - 예 : ActionEvent, AdjustmentEvent, ItemEvent, TextEvent 등

- 저수준 이벤트
  - 의미 이벤트를 가능하게 하는 이벤트를 의미(의미 이벤트인 버튼 클릭은 마우스 이동, 마우스 누름, 마우스 놓기 등 여러 단계의 세부적인 이벤트로 구성)
  - 모든 스윙 컴포넌트에서 발생
  - 예 : ComponentEvent, ContainerEvent, FocusEvent, MouseEvent, KeyEvent 등

이벤트 객체와 이벤트 소스(1)

이벤트 객체	이벤트 소스	이벤트가 발생하는 경우
ActionEvent	JButton	마우스로 버튼을 클릭하거나 키로 버튼을 선택한 경우
	JComboBox	아이템을 선택한 경우
	JList	리스트 아이템을 더블클릭하여 리스트 아이템을 선택한 경우
	JMenuItem	메뉴 아이템 선택을 선택한 경우
	JTextField	텍스트 입력 중 <Enter> 키를 누른 경우
ItemEvent	JCheckBox	체크박스의 선택 혹은 해제
	JCheckBoxMenuItem	체크박스 메뉴 아이템이 선택 혹은 해제 될 때
	JComboBox	아이템이 선택 되거나 해제되었을 때
	JList	리스트 아이템이 선택될 때
KeyEvent	Component	모든 컴포넌트에 대해, 키가 눌러지거나 눌러진 키가 떼어질 때
MouseEvent	Component	모든 컴포넌트에 대해, 마우스 버튼이 눌러지거나 떼어질 때, 클릭될 때, 컴포넌트 위에 마우스가 올라갈 때, 올라간 마우스가 내려올 때, 마우스가 드래그될 때, 마우스가 단순 움직일 때

이벤트 객체와 이벤트 소스(2)

이벤트 객체	이벤트 소스	이벤트가 발생하는 경우
FocusEvent	Component	모든 컴포넌트에 대해, 컴포넌트가 포커스를 받거나 잃을 때
TextEvent	TextField	텍스트가 변경될 때
	TextArea	텍스트가 변경될 때
WindowEvent	Window	Window를 상속받는 모든 컴포넌트에 대해, 윈도우가 활성화, 비활성화, 아이콘화, 아이콘에서 복구 될 때, 윈도우가 열리거나 닫힐 때, 윈도우가 종료될 때 등
AdjustmentEvent	JScrollBar	스크롤바를 사용자가 움직였을 때
ComponentEvent	Component	모든 컴포넌트에 대해, 컴포넌트가 사라지거나, 나타나거나, 이동 하거나 크기 변경 될 때
ContainerEvent	Container	Container에 컴포넌트가 추가 혹은 삭제되었을 때

이벤트 리스너

- 이벤트 리스너란?
  - 이벤트를 처리하는 코드, 클래스로 작성
- JDK에서 이벤트 리스너 작성을 위한 인터페이스(interface) 제공
  - 개발자가 리스너 인터페이스의 추상 메소드 구현
    - 이벤트가 발생하면 자바 플랫폼은 리스너 인터페이스의 추상 메소드 호출

- 이벤트 소스에 이벤트 리스너 등록

```
void addXXXListener(이벤트리스너객체);
```

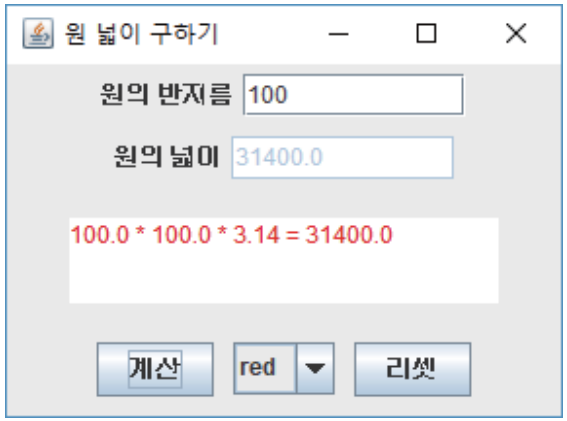
컴포넌트에서 발생하는 이벤트 이름이다. ActionEvent면 XXX는 Action이며, MouseEvent면 XXX는 Mouse이다.

이벤트 리스너 작성 방법

- 독립 클래스로 작성
  - 이벤트 리스너를 완전한 클래스로 작성
  - 이벤트 리스너를 여러 곳에서 사용할 때 적합
- 내부 클래스(inner class)로 작성
  - 클래스 안에 멤버처럼 클래스 작성
  - 이벤트 리스너를 특정 클래스에서만 사용할 때 적합
- 익명 클래스(anonymous class)로 작성
  - 클래스의 이름 없이 간단히 리스너 작성
  - 클래스 조차 만들 필요 없이 리스너 코드가 간단한 경우에 적합
- 람다식으로 작성
  - 리스너가 함수형 인터페이스인 경우

원의 넓이 구하기

```
1  package sec03.not;
2
3  import java.awt.BorderLayout;
15
16 public class EventDemo extends JFrame {
17     JTextField t1, t2;
18     JTextArea area;
19     JButton cal, reset;
20     JComboBox<String> cb;
21
22     EventDemo() {
23         setTitle("원 넓이 구하기");
24
25         setLayout(new BorderLayout(10, 10));
26         showNorth();
27         showCenter();
28         showSouth();
29
30         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
31         setSize(300, 220);
32         setVisible(true);
33     }
```



```
35      void showNorth() {
36          JPanel p1 = new JPanel();
37          JPanel p2 = new JPanel();
38          JPanel panel = new JPanel(new GridLayout(2, 0));
39
40          JLabel l1 = new JLabel("원의 반지름");
41          JLabel l2 = new JLabel("원의 넓이");
42
43          t1 = new JTextField(10);
44          t2 = new JTextField(10);
45          t2.setEnabled(false);
46
47          p1.add(l1);
48          p1.add(t1);
49          p2.add(l2);
50          p2.add(t2);
51          panel.add(p1);
52          panel.add(p2);
53
54          add(panel, BorderLayout.NORTH);
55      }
56
57      void showCenter() {
58          JPanel panel = new JPanel();
59
60          area = new JTextArea(30, 20);
61          area.setText("이 영역에 원의 넓이를\n계산하는 과정이 나타납니다.");
62          area.setEditable(false);
63          area.setForeground(Color.RED);
64
65          panel.add(area);
66
67          add(panel, BorderLayout.CENTER);
68      }
```

```
70      void showSouth() {
71          String[] color = { "red", "blue" };
72
73          JPanel panel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 10));
74
75          cal = new JButton("계산");
76          cb = new JComboBox<>(color);
77          reset = new JButton("리셋");
78
79          panel.add(cal);
80          panel.add(cb);
81          panel.add(reset);
82
83          add(panel, BorderLayout.SOUTH);
84      }
85
86      public static void main(String[] args) {
87          new EventDemo();
88      }
89  }
```

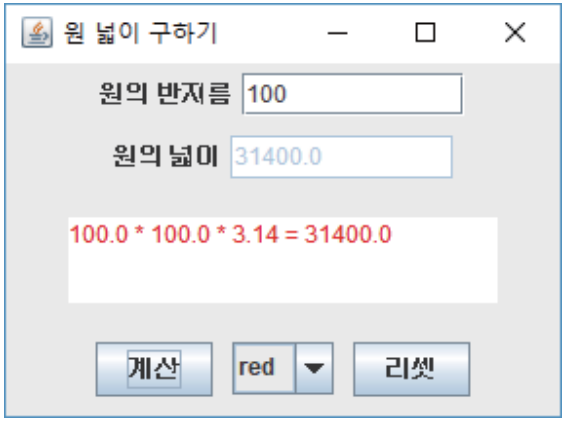
ActionEvent 처리를 추가한 원 넓이 구하기(1)

```
67    void showSouth() {
...       // 생략
83    ActionListener listener1 = e -> {
84       if (e.getSource() == cal) {
85          if (t1.getText().isEmpty())
86             area.setText("반지름을 입력하세요!!!");
87          else {
88             String s = t1.getText();
89             double radius = Double.parseDouble(s);
90             double result = radius * radius * 3.14;
91             t2.setText("" + result);
92             area.setText(radius + " * " + radius + " * 3.14 = " + result);
93          }
94       } else {
```

ActionListener 객체를 생성한다.

ActionEvent의 소스가 계산 버튼인지 조사한다.

반지름 텍스트 필드에 입력 데이터가 있는지 조사한다. 입력 데이터가 없다면 오류 메시지를 텍스트 영역에 추가한다. 입력 데이터가 있다면 넓이를 계산한 후 넓이 텍스트 필드에 추가하고, 계산 과정도 텍스트 영역에 추가한다.



ActionEvent 처리를 추가한 원 넓이 구하기(2)

```
95            t1.setText("");
96            t2.setText("");
97            area.setText("");
98         }
99      };
100
101     cal.addActionListener(listener1);
102     reset.addActionListener(listener1);
103  }
104
105  public static void main(String[] args) {
106     new EventDemo();
107  }
108 }
```

반지름과 넓이의 텍스트 필드와 텍스트 영역에 있는 내용을 지운다.

버튼에 리스너를 등록한다.

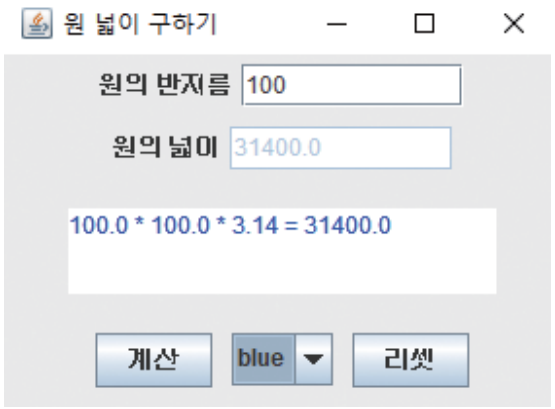
ItemEvent 처리를 추가한 원 넓이 구하기

```
69   void showSouth() {
...     // 생략
103     cb.addItemListener(e -> {
104        int index = ((JComboBox) cb).getSelectedIndex();
105        if (index == 0)
106           area.setForeground(Color.RED);
107        else
108           area.setForeground(Color.BLUE);
109     });
110  }
```

콤보 박스에 ItemListener 객체를 람다식으로 등록한다.

선택한 콤보 박스 항목의 인덱스를 가져온다.

콤보 박스의 선택한 항목에 따라 글자색을 변경한다.



KeyEvent 처리를 추가한 원 넓이 구하기

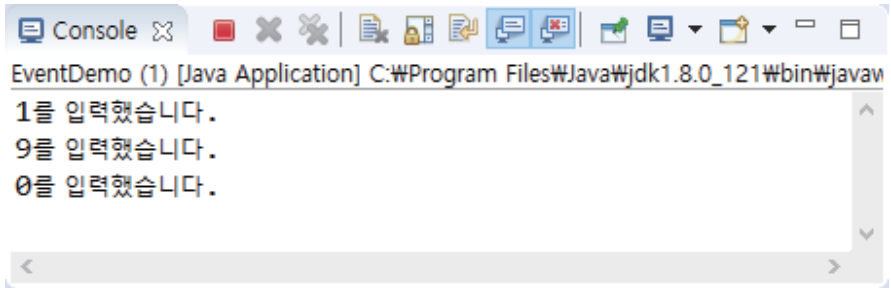
```
36    void showNorth() {
...       // 생략
57        KeyListener listener2 = new KeyListener() {
58            public void keyTyped(KeyEvent e) {
59                System.out.println(e.getKeyChar() + "를 입력했습니다.");
60            }
61
62            public void keyReleased(KeyEvent e) {
63            }
64
65            public void keyPressed(KeyEvent e) {
66            }
67        };
68        t1.addKeyListener(listener2);
69    }
```

KeyListener 객체를 생성한다.

키보드로 입력하면 입력한 문자를 콘솔 뷰에 출력한다.

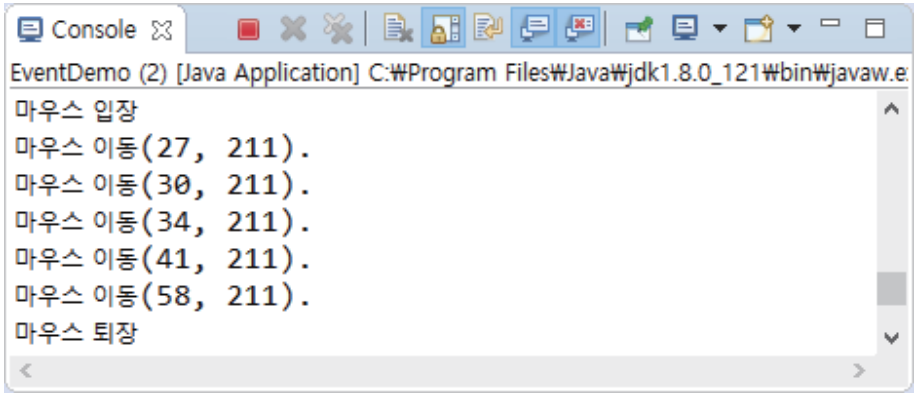
KeyListener 인터페이스에 있는 추상 메서드이므로 사용하지 않아도 필요하다.

텍스트 필드에 생성된 KeyListener 객체를 등록한다.



MouseEvent 처리를 추가한 원 넓이 구하기(1)

```
addMouseListener(new MouseListener() {
    public void mouseClicked(MouseEvent e) {
        System.out.println("마우스 클릭");
    }

    public void mousePressed(MouseEvent e) {
        System.out.println("마우스 버튼 누르기");
    }

    public void mouseReleased(MouseEvent e) {
        System.out.println("마우스 버튼 놓기");
    }

    public void mouseEntered(MouseEvent e) {
        System.out.println("마우스 입장");
    }

    public void mouseExited(MouseEvent e) {
        System.out.println("마우스 퇴장");
    }
});
```



MouseEvent 처리를 추가한 원 넓이 구하기(2)

```
60      addMouseMotionListener(new MouseMotionListener() {
61         public void mouseDragged(MouseEvent e) {
62            System.out.println("마우스 드래그(" + e.getX() + ", "
                        + e.getY() + ").");
63         }
64
65         public void mouseMoved(MouseEvent e) {
66            System.out.println("마우스 이동(" + e.getX() + ", "
                        + e.getY() + ").");
67         }
68      });
69   }
...  // 생략
```

MouseMotionListener 무명 객체를
이벤트 소스인 프로그램 자신에 등록한다.

어댑터 클래스 기초(1)

- 개발자가 필요한 추상 메서드만 구현하면 되도록 리스너에 대응하는 어댑터 클래스를 제공
- 어댑터 클래스는 리스너 인터페이스에 포함된 모든 추상 메서드를 빈 본체로 구현한 클래스에 불과
- 예 :

```
public abstract class KeyAdapter implements KeyListener {
  void keyPressed(KeyEvent e) { }
  void keyReleased(KeyEvent e) { }
  void keyTyped(KeyEvent e) { }
}
```

본체는 비어 있다.

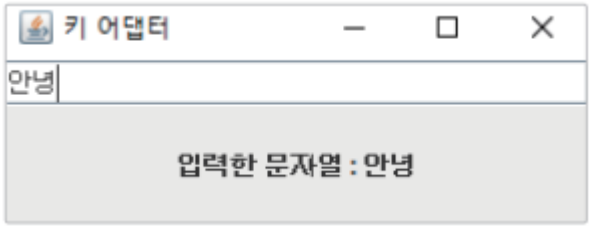
어댑터 클래스 기초(2)

- 리스너 인터페이스와 대응하는 어댑터 클래스

리스너 인터페이스	어댑터 클래스
ComponentListener	ComponentAdapter
ContainerListener	ContainerAdapter
FocusListener	FocusAdapter
KeyListener	KeyAdapter
MouseListener	MouseAdapter
MouseMotionListener	MouseMotionAdapter
WindowListener	WindowAdapter

어댑터 클래스 기초(3)

- KeyAdapter 클래스
  - 예제 : [예제 15-8]



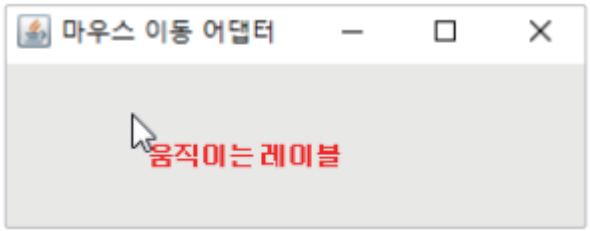

```
18      t.addKeyListener(new KeyAdapter() {
19         public void keyPressed(KeyEvent e) {
20            if (e.getKeyCode() == KeyEvent.VK_ENTER) {
21               l.setText("입력한 문자열 : " + t.getText());
22            }
23         }
24      });
```

KeyAdapter의 무명 자식 객체이다. KeyListener와는 달리 필요한 메서드만 오버라이딩하면 된다.

```java
package sec04;

import java.awt.event.KeyAdapter;

public class KeyAdapterDemo extends JFrame {
    public KeyAdapterDemo() {
        setTitle("키 어댑터");

        JLabel l = new JLabel("", JLabel.CENTER);
        JTextField t = new JTextField(10);

        add("North", t);
        add("Center", l);

        t.addKeyListener(new KeyAdapter() {
            public void keyPressed(KeyEvent e) {
                if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                    l.setText("입력한 문자열 : " + t.getText());
                }
            }
        });

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 120);
        setVisible(true);
    }

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

어댑터 클래스

- MouseMotionAdapter 클래스
  - 예제 : [예제 15-9]




```
12      JLabel label = new JLabel("움직이는 레이블");
13      label.setForeground(Color.RED);
14      add(label);

16      addMouseMotionListener(new MyMouseMotionAdapter(label));
```

```
28  class MyMouseMotionAdapter extends MouseMotionAdapter {
29    JLabel label;
30
31    public MyMouseMotionAdapter(JLabel label) {
32       this.label = label;
33    }
34
35    public void mouseMoved(MouseEvent e) {
36       label.setLocation(e.getX(), e.getY() - 50);
37    }
38  }
```

마우스 이동 이벤트만 처리하려고 mouseMoved( ) 메서드만 오버로딩한다.

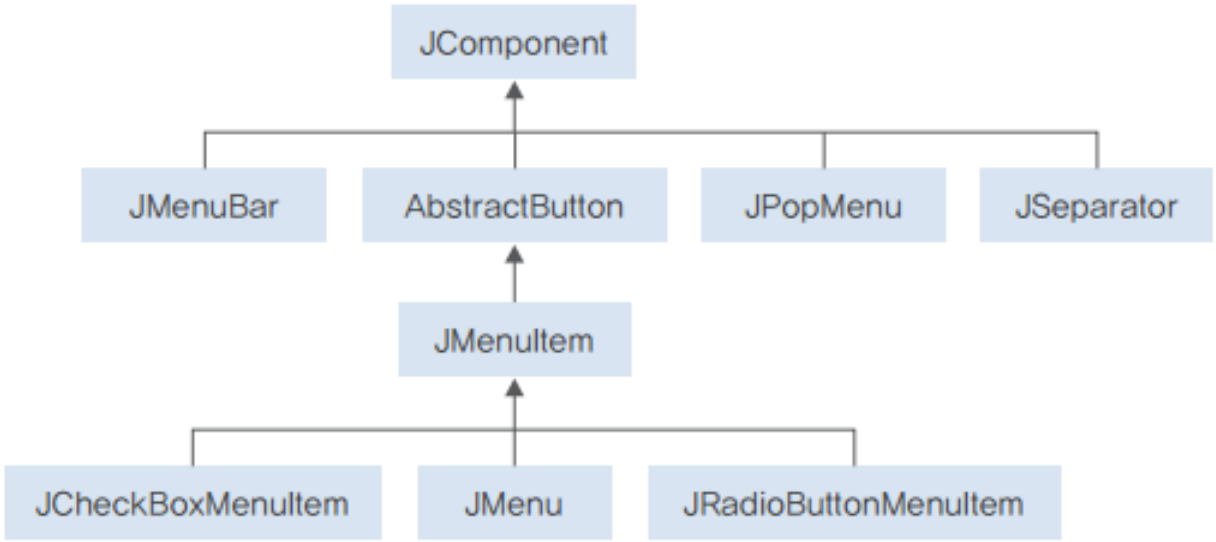
마우스 커서의 위치를 보정하려고 -50을 했다.

```java
package sec04;

import java.awt.Color;

public class MouseMotionAdapterDemo extends JFrame {
    MouseMotionAdapterDemo() {
        setTitle("마우스 이동 어댑터");

        JLabel label = new JLabel("움직이는 레이블");
        label.setForeground(Color.RED);
        add(label);

        addMouseMotionListener(new MyMouseMotionAdapter(label));

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 120);
        setVisible(true);
    }

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

class MyMouseMotionAdapter extends MouseMotionAdapter {
    JLabel label;

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

    public void mouseMoved(MouseEvent e) {
        label.setLocation(e.getX(), e.getY() - 50);
    }
}
```

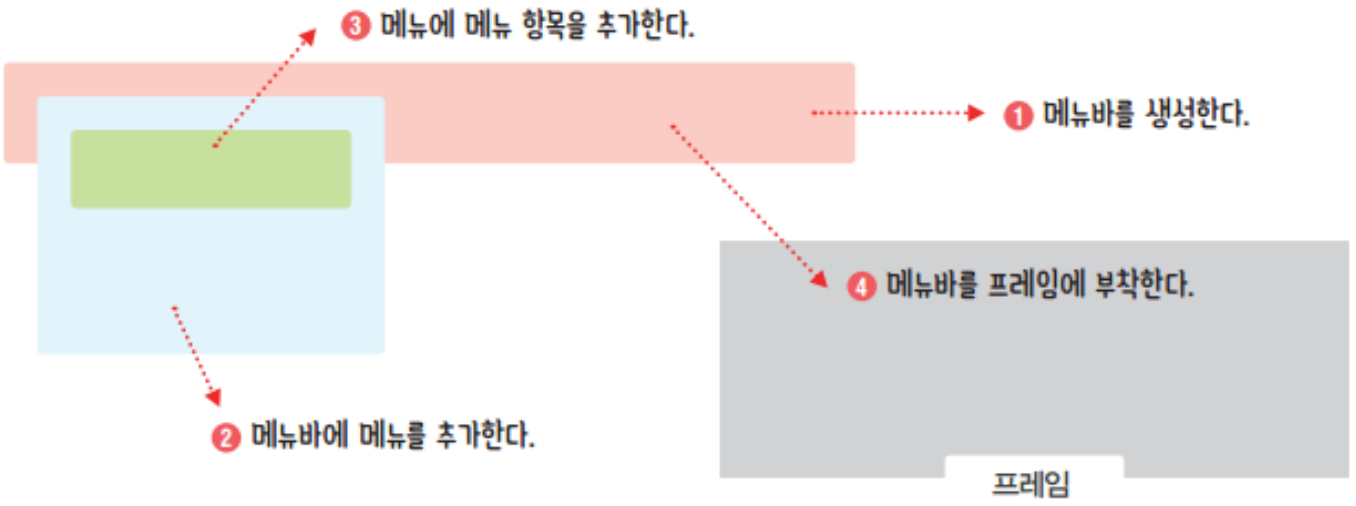
이벤트와 메뉴 기초

- 메뉴의 구성

- 메뉴의 계층 구조



메뉴 구성 순서



❶ 메뉴 바 생성

```
JMenuBar mb = JMenuBar();
```

❷ 메뉴바에 메뉴 추가

```
JMenu menu = new JMenu("File");
mb.add(menu);
```

❸ 메뉴에 메뉴 항목 추가

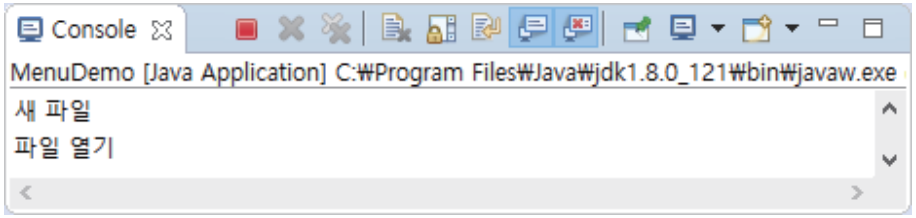
```
JMenuItem item = new JMenuItem("New File");
menu.add(item);
```

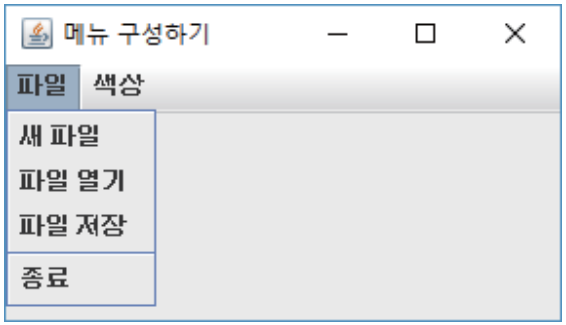
❹ 프레임에 메뉴바 부착

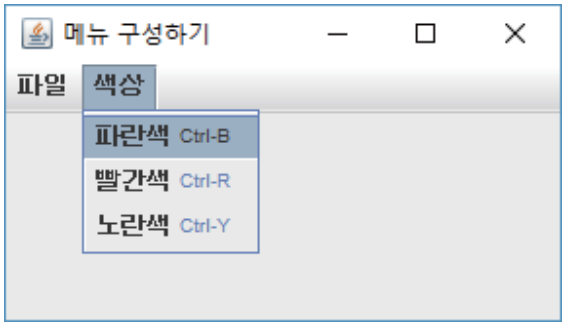
```
frame.setJMenuBar(mb);
```

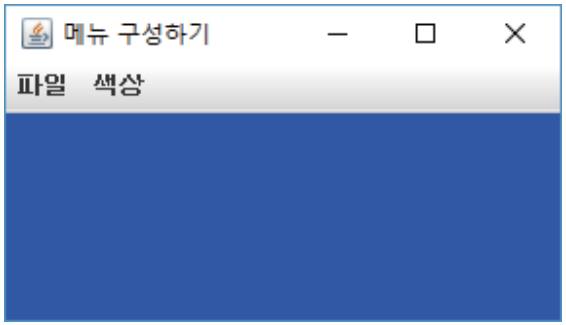
이벤트와 메뉴 응용

- 예제 : [예제 15-10]









```java
package sec05;

import java.awt.Color;

public class MenuDemo extends JFrame implements ActionListener {
    MenuDemo() {
        setTitle("메뉴 구성하기");
        makeMenu();
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 170);
        setVisible(true);
    }

    void makeMenu() {
        JMenuItem item;
        KeyStroke key;

        JMenuBar mb = new JMenuBar();
        JMenu m1 = new JMenu("파일");
        m1.setMnemonic(KeyEvent.VK_F);
        JMenu m2 = new JMenu("색상");
        m2.setMnemonic(KeyEvent.VK_C);

        item = new JMenuItem("새 파일", KeyEvent.VK_N);
        item.addActionListener(this);
        m1.add(item);
        item = new JMenuItem("파일 열기", KeyEvent.VK_O);
        item.addActionListener(this);
        m1.add(item);
        m1.add(new JMenuItem("파일 저장"));
        m1.addSeparator();
        m1.add(new JMenuItem("종료"));
```

```
43          item = new JMenuItem("파란색");
44          key = KeyStroke.getKeyStroke(KeyEvent.VK_B, ActionEvent.CTRL_MASK);
45          item.setAccelerator(key);
46          item.addActionListener(this);
47          m2.add(item);
48          item = new JMenuItem("빨간색");
49          key = KeyStroke.getKeyStroke(KeyEvent.VK_R, ActionEvent.CTRL_MASK);
50          item.setAccelerator(key);
51          item.addActionListener(this);
52          m2.add(item);
53          item = new JMenuItem("노란색");
54          key = KeyStroke.getKeyStroke(KeyEvent.VK_Y, ActionEvent.CTRL_MASK);
55          item.setAccelerator(key);
56          item.addActionListener(this);
57          m2.add(item);
58          mb.add(m1);
59          mb.add(m2);
60          setJMenuBar(mb);
61      }
62
63      public static void main(String[] args) {
64          new MenuDemo();
65      }
```

```
67      public void actionPerformed(ActionEvent e) {
68          JMenuItem mi = (JMenuItem) (e.getSource());
69
70          switch (mi.getText()) {
71          case "새 파일":
72              System.out.println("새 파일");
73              break;
74          case "파일 열기":
75              System.out.println("파일 열기");
76              break;
77          case "파란색":
78              this.getContentPane().setBackground(Color.BLUE);
79              break;
80          case "빨간색":
81              this.getContentPane().setBackground(Color.RED);
82              break;
83          case "노란색":
84              this.getContentPane().setBackground(Color.YELLOW);
85          }
86      }
87  }
```