

11

기본적인 스윙 컴포넌트와 활용

GUI를 구성하는 2 가지 방법

2

1. 컴포넌트 기반 GUI 프로그래밍

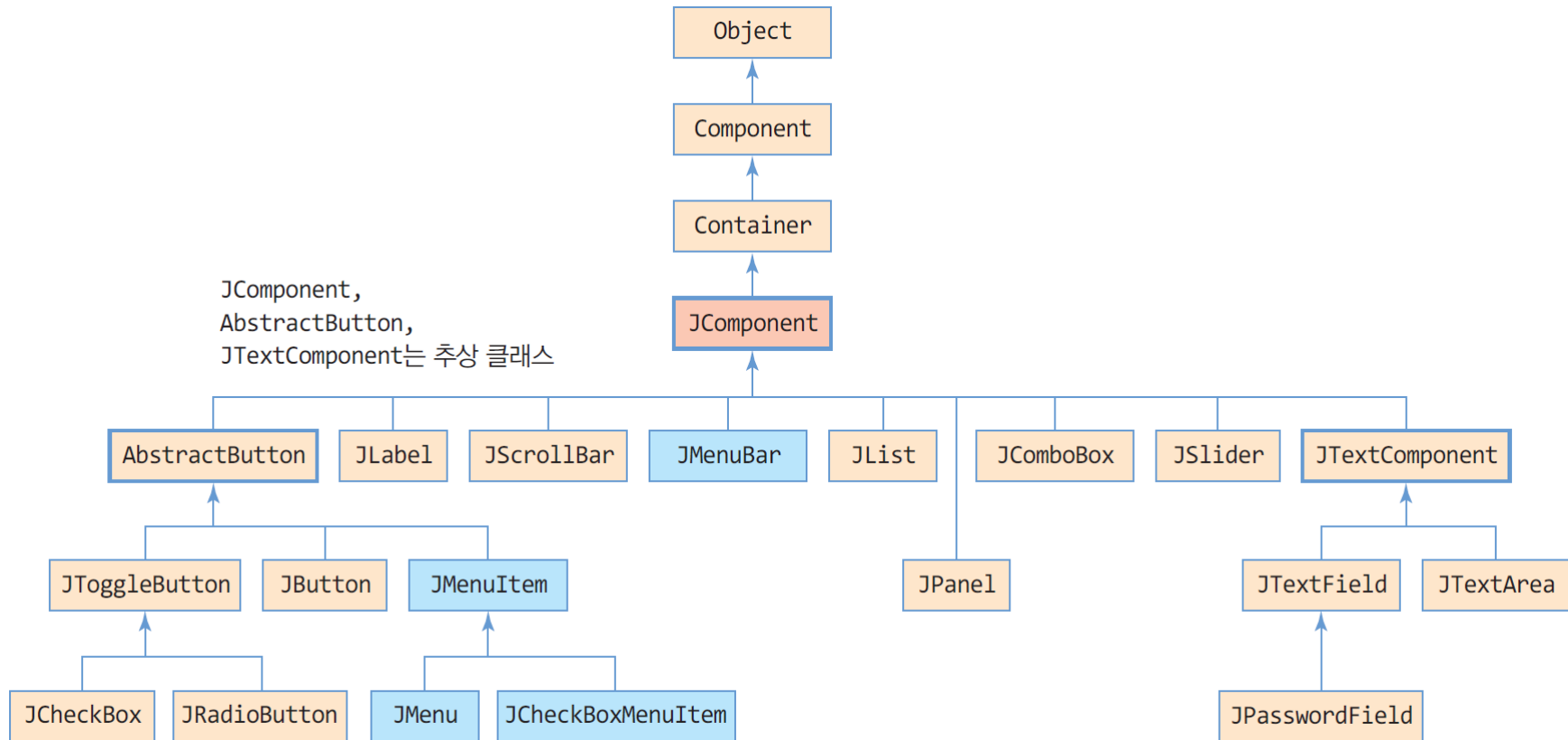
- ▣ 스윙 패키지에 주어진 GUI 컴포넌트 이용
- ▣ GUI 구성이 쉽다.
- ▣ 자바 패키지에 제공하는 GUI 컴포넌트 한계
- ▣ 일반적인 GUI 프로그램에 적합

2. 그래픽 기반 GUI 프로그래밍

- ▣ 선, 원, 도형, 이미지를 직접 그리는 그래픽 화면 구성
 - ▣ 개발자의 작업 부담 높음
 - ▣ 자바 패키지에 없는 독특한 GUI 구성 가능
 - ▣ 게임 등 자유로운 GUI
- 11장 : 컴포넌트 기반 GUI 프로그래밍
 - 12장 : 그래픽 기반 GUI 프로그래밍

기초적인 스윙 컴포넌트와 상속 관계

3



스윙 컴포넌트의 공통 메소드, JComponent의 메소드

4

컴포넌트의 모양과 관련된 메소드

```
void setForeground(Color)  전경색 설정  
void setBackground(Color)  배경색 설정  
void setOpaque(boolean)  불투명성 설정  
void setFont(Font)  폰트 설정  
Font getFont()  폰트 리턴
```

컴포넌트의 상태와 관련된 메소드

```
void setEnabled(boolean)  컴포넌트 활성화/비활성화  
void setVisible(boolean)  컴포넌트 보이기/숨기기  
boolean isVisible()  컴포넌트의 보이는 상태 리턴
```

컴포넌트의 위치와 크기에 관련된 메소드

```
int getWidth()  폭 리턴  
int getHeight()  높이 리턴  
int getX()  x 좌표 리턴  
int getY()  y 좌표 리턴  
Point getLocationOnScreen()  스크린 좌표상에서의 컴포넌트 좌표  
void setLocation(int, int)  위치 지정  
void setSize(int, int)  크기 지정
```

컨테이너를 위한 메소드

```
Component add(Component)  자식 컴포넌트 추가  
void remove(Component)  자식 컴포넌트 제거  
void removeAll()  모든 자식 컴포넌트 제거  
Component[] getComponents()  자식 컴포넌트 배열 리턴  
Container getParent()  부모 컨테이너 리턴  
Container getTopLevelAncestor()  최상위 부모 컨테이너 리턴
```

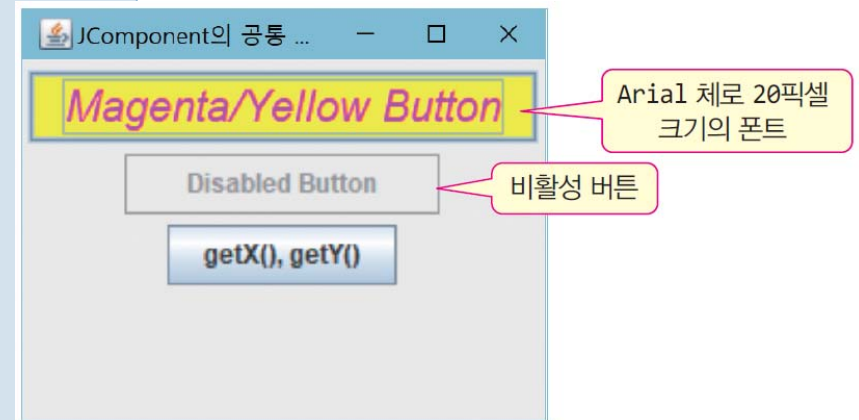
예제 11-1 : 스윙 컴포넌트의 공통 기능, JComponent의 메소드

5

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

```
public class JComponentEx extends JFrame {
    public JComponentEx() {
        super("JComponent의 공통 메소드 예제");
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        JButton b1 = new JButton("Magenta/Yellow Button");
        JButton b2 = new JButton(" Disabled Button ");
        JButton b3 = new JButton("getX(), getY()");

        b1.setBackground(Color.YELLOW); // 배경색 설정
        b1.setForeground(Color.MAGENTA); // 글자색 설정
        b1.setFont(new Font("Arial", Font.ITALIC, 20)); // Arial, 20픽셀 폰트 설정
        b2.setEnabled(false); // 버튼 비활성화
        b3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JButton b = (JButton)e.getSource();
                JComponentEx frame = (JComponentEx)b.getTopLevelAncestor();
                frame.setTitle(b.getX() + "," + b.getY()); // 타이틀에 버튼 좌표 출력
            }
        });
        c.add(b1); c.add(b2); c.add(b3); // 컨테이너에 버튼 부착
        setSize(260,200);
        setVisible(true);
    }
    public static void main(String[] args) {
        new JComponentEx();
    }
}
```



JLabel, 레이블 컴포넌트

6

- JLabel의 용도
 - ▣ 문자열이나 이미지를 컴포넌트화 하여 출력하기 위한 목적
- 생성자

`JLabel()` 빈 레이블

`JLabel(Icon image)` 이미지 레이블

`JLabel(String text)` 문자열 레이블

`JLabel(String text, Icon image, int hAlign)` 문자열과 이미지를 모두 가진 레이블

- hAlign: 수평 정렬 값으로 `SwingConstants.LEFT`, `SwingConstants.RIGHT`, `SwingConstants.CENTER` 중 하나

레이블 컴포넌트 생성 예

7

□ 단순 텍스트 만을 가진 레이블 컴포넌트 생성

```
JLabel textLabel = new JLabel("사랑합니다");
```

□ 이미지를 가진 레이블 컴포넌트 생성

- 이미지 파일로부터 이미지를 읽기 위해 ImageIcon 클래스 사용
- 다룰 수 있는 이미지 : png, gif, jpg
 - sunset.jpg의 경로명이 "images/sunset.jpg"인 경우

```
ImageIcon image = new ImageIcon("images/sunset.jpg");  
JLabel imageLabel = new JLabel(image);
```

□ 수평 정렬 값을 가진 레이블 컴포넌트 생성

- 텍스트 이미지 모두 출력하고자 하는 경우 수평 정렬 지정

```
ImageIcon image = new ImageIcon("images/sunset.jpg");  
JLabel label = new JLabel("사랑합니다", image, SwingConstants.CENTER);
```

예제 11-2 : JLabel을 이용한 레이블 만들기

8

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

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

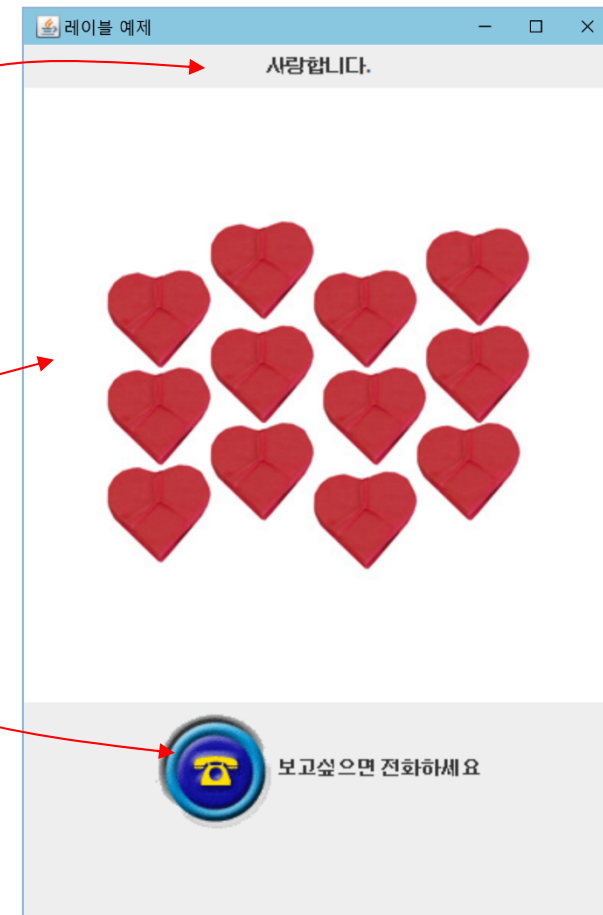
        JLabel textLabel = new JLabel("사랑합니다.");

        ImageIcon beauty = new ImageIcon("images/beauty.jpg");
        JLabel imageLabel = new JLabel(beauty);

        ImageIcon normalIcon = new ImageIcon("images/normalIcon.gif");
        JLabel label = new JLabel("보고싶으면 전화하세요",
                                   normalIcon, SwingConstants.CENTER);

        c.add(textLabel);
        c.add(imageLabel);
        c.add(label);

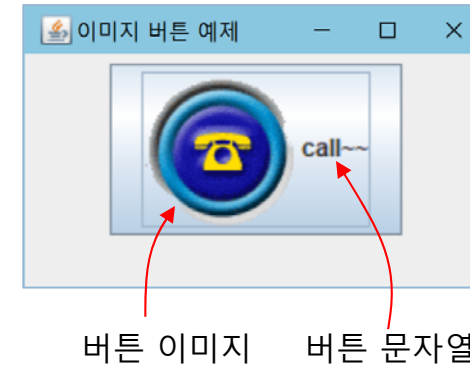
        setSize(400,600);
        setVisible(true);
    }
    public static void main(String [] args) {
        new LabelEx();
    }
}
```



JButton, 버튼 컴포넌트

9

- 버튼 컴포넌트
 - ▣ 버튼 모양의 컴포넌트
 - ▣ 버튼을 선택하면 Action 이벤트 발생
- 생성자



```
JButton() 빈 버튼  
JButton(Icon image) 이미지 버튼  
JButton(String text) 문자열 버튼  
JButton(String text, Icon image) 문자열과 이미지를 가진 버튼
```

- 버튼 컴포넌트 생성 예
 - ▣ "hello" 문자열을 가진 버튼 컴포넌트 생성 예

```
JButton btn = new JButton("hello");
```

이미지 버튼 만들기

10

- 하나의 버튼에 3 개의 이미지 연결
 - ▣ 마우스 접근에 따라 서로 다른 3 개의 이미지 출력 가능
 - ▣ 사용자의 버튼 조작에 대한 시각적 효과를 극대화

- 3 개의 버튼 이미지
 1. 버튼의 보통 상태 때 출력되는 이미지
 - 생성자에 이미지 아이콘 전달
 - 이미지 설정 메소드 : JButton의 setIcon(Image image)
 2. 버튼에 마우스가 올라갈 때 출력되는 이미지
 - 이미지 설정 메소드 : JButton.setRolloverIcon(Image);
 3. 버튼을 누르고 있는 동안 출력되는 이미지
 - 이미지 설정 메소드 : JButton.setPressedIcon(Image)

- 이미지 아이콘 생성
 - ▣ new ImageIcon(이미지 경로명);
 - 예) new ImageIcon("images/normalIcon.gif");

예제 11-3 : JButton을 이용한 버튼 만들기

11

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

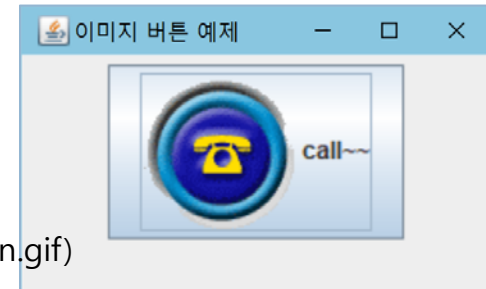
public class ButtonEx extends JFrame {
    public ButtonEx() {
        setTitle("이미지 버튼 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        ImageIcon normalIcon = new ImageIcon("images/normalIcon.gif");
        ImageIcon rolloverIcon = new ImageIcon("images/rolloverIcon.gif");
        ImageIcon pressedIcon = new ImageIcon("images/pressedIcon.gif");

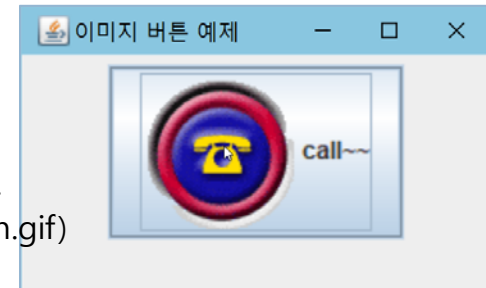
        JButton btn = new JButton("call~~", normalIcon);
        btn.setPressedIcon(pressedIcon);
        btn.setRolloverIcon(rolloverIcon);

        c.add(btn);
        setSize(250,150);
        setVisible(true);
    }
    public static void main(String [] args) {
        new ButtonEx();
    }
}
```

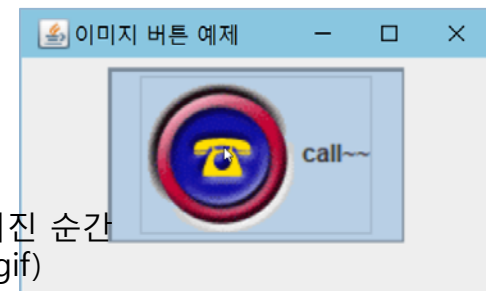
보통 상태
(normalIcon.gif)



마우스가
올라간 경우
(rolloverIcon.gif)



마우스가 눌려진 순간
(pressedIcon.gif)



레이블과 버튼의 정렬(Alignment)

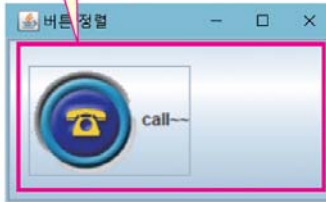
12

- 수평 정렬 : 컴포넌트 내에 이미지와 텍스트의 수평 위치

```
void setHorizontalAlignment(int align)
```

• align: 정렬의 기준을 지정하는 값으로 다음과 같다.

SwingConstants.LEFT, SwingConstants.CENTER, SwingConstants.RIGHT



왼쪽 정렬
SwingConstants.LEFT



중앙 정렬
SwingConstants.CENTER



오른쪽 정렬
SwingConstants.RIGHT

- 수직 정렬 : 컴포넌트 내에 이미지와 텍스트의 수직 위치

```
void setVerticalAlignment(int align)
```

• align: 정렬의 기준을 지정하는 값으로 다음과 같다.

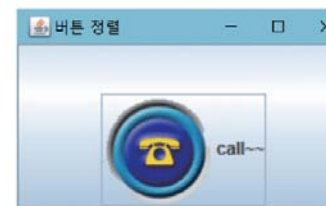
SwingConstants.TOP, SwingConstants.CENTER, SwingConstants.BOTTOM



위쪽 정렬
SwingConstants.TOP



중앙 정렬
SwingConstants.CENTER



아래쪽 정렬
SwingConstants.BOTTOM

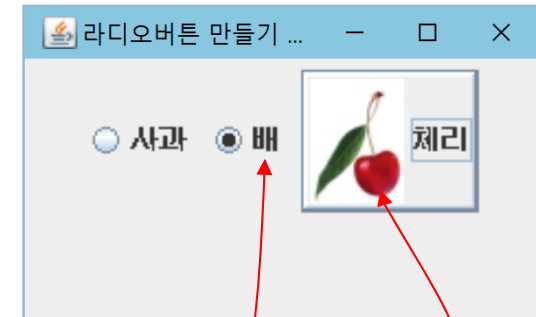
JCheckBox, 체크박스 컴포넌트

13

□ JCheckBox

- 선택(selected)과 비선택(deselected)의 두 상태만 가지는 체크 버튼

□ 생성자



체크박스 문자열

체크박스 이미지

`JCheckBox()` 빈 체크박스

`JCheckBox(String text)` 문자열 체크박스

`JCheckBox(String text, boolean selected)` 문자열 체크박스

`JCheckBox(Icon image)` 이미지 체크박스

`JCheckBox(Icon image, boolean selected)` 이미지 체크박스

`JCheckBox(String text, Icon image)` 문자열과 이미지를 가진 체크박스

`JCheckBox(String text, Icon image, boolean selected)` 문자열과 이미지를 가진 체크박스

• selected: true이면 선택 상태로 초기화. 디폴트는 해제 상태

체크 박스 생성

14

□ 문자열 체크 박스

- "사과" 텍스트를 가진 체크박스 생성

```
JCheckBox c = new JCheckBox("사과");
```

- "배" 텍스트를 가지고 선택 상태로 체크박스 생성

```
JCheckBox c = new JCheckBox("배", true);
```

- 체크 박스 모양 ☒이 명료하게 출력되고 사용자는 이것을 체크

□ 이미지 아이콘을 가진 체크 박스 생성 예

- 체크 박스 모양 ☒이 출력되지 않음

- 선택 상태를 표현하는 이미지 아이콘을 따로 지정해야 함

- cherry.jpg 이미지와 "체리" 텍스트를 가진 체크 박스 생성 예

```
ImageIcon cherryIcon = new ImageIcon("images/cherry.jpg");  
ImageIcon selectedCherryIcon = new  
ImageIcon("images/selectedCherry.jpg");  
JCheckBox cherry = new JCheckBox("체리", cherryIcon);  
cherry.setSelectedIcon(selectedCherryIcon); // 선택 상태의 이미지 달기
```


예제 11-4 : 체크박스 생성 예

15

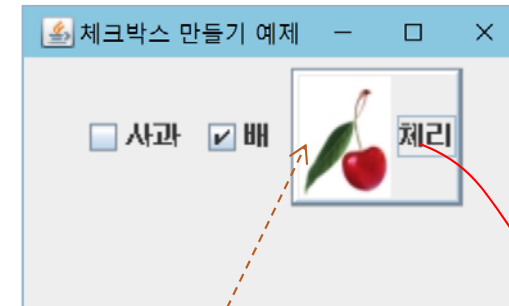
```
import javax.swing.*;
import java.awt.*;

public class CheckBoxEx extends JFrame {
    public CheckBoxEx() {
        setTitle("체크박스 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        ImageIcon cherryIcon = new ImageIcon("images/cherry.jpg");
        ImageIcon selectedCherryIcon = new ImageIcon(
            "images/selectedCherry.jpg");

        JCheckBox apple = new JCheckBox("사과");
        JCheckBox pear = new JCheckBox("배", true);
        JCheckBox cherry = new JCheckBox("체리", cherryIcon);
        cherry.setBorderPainted(true);
        cherry.setSelectedIcon(selectedCherryIcon);

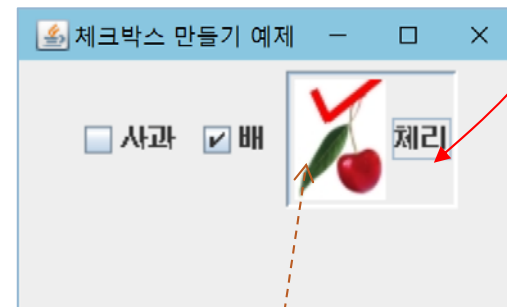
        c.add(apple);
        c.add(pear);
        c.add(cherry);

        setSize(250,150);
        setVisible(true);
    }
    public static void main(String [] args) {
        new CheckBoxEx();
    }
}
```



cherry.jpg(선택되지 않은 상태)

체크 박스를
선택하면



selectedCherry.jpg(선택된 상태)

JCheckBox에서 Item 이벤트 처리

16

□ Item 이벤트

- 체크 박스나 라디오버튼의 선택 상태가 바뀔 발생하는 이벤트
 - 마우스나 키보드로 체크박스를 선택 상태를 바꾸는 경우
 - 프로그램에서 선택 상태를 바꾸는 경우

```
JCheckBox c = new JCheckBox("사과");  
c.setSelected(true); // 선택 상태 변경
```

□ ItemListener 인터페이스의 추상 메소드

```
void itemStateChanged(ItemEvent e) 체크박스의 선택/해제 상태가 변하는 경우 호출
```

□ ItemEvent의 주요 메소드

```
int getStateChange()
```

리턴 값은 선택된 경우 `ItemEvent.SELECTED`, 해제된 경우 `ItemEvent.DESELECTED`

```
Object getItem()
```

이벤트를 발생시킨 아이템 객체 리턴. 체크박스의 경우 `JCheckBox` 컴포넌트의 레퍼런스 리턴

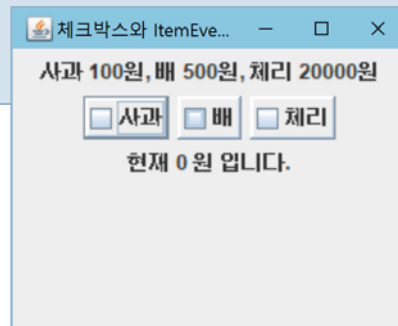
예제 11-5 : ItemEvent 활용하여 체크박스로 가격 합산하기

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

public class CheckBoxItemEventEx extends JFrame {
    private JCheckBox [] fruits = new JCheckBox [3];
    private String [] names = {"사과", "배", "체리"};
    private JLabel sumLabel;

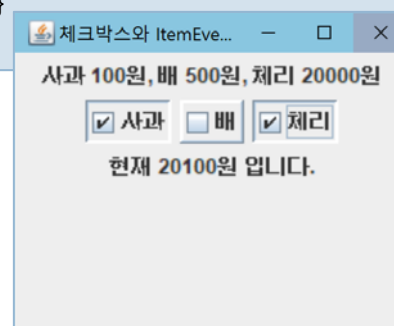
    public CheckBoxItemEventEx() {
        setTitle("체크박스와 ItemEvent 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        c.add(new JLabel("사과 100원, 배 500원, 체리 20000원"));

        MyItemListener listener = new MyItemListener();
        for(int i=0; i<fruits.length; i++) {
            fruits[i] = new JCheckBox(names[i]);
            fruits[i].setBorderPainted(true);
            c.add(fruits[i]);
            fruits[i].addItemListener(listener);
        }
        sumLabel = new JLabel("현재 0 원 입니다.");
        c.add(sumLabel);
        setSize(250,200);
        setVisible(true);
    }
}
```



```
class MyItemListener implements ItemListener {
    private int sum = 0; // 가격의 합
    public void itemStateChanged(ItemEvent e) {
        if(e.getStateChange() == ItemEvent.SELECTED) {
            if(e.getItem() == fruits[0])
                sum += 100;
            else if(e.getItem() == fruits[1])
                sum += 500;
            else
                sum += 20000;
        }
        else {
            if(e.getItem() == fruits[0])
                sum -= 100;
            else if(e.getItem() == fruits[1])
                sum -= 500;
            else
                sum -= 20000;
        }
        sumLabel.setText("현재 "+ sum + "원 입니다.");
    }
}

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

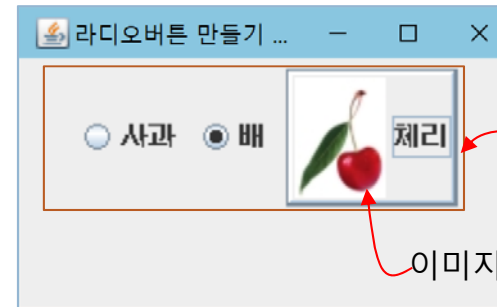


JRadioButton, 라디오버튼 컴포넌트

18

□ JRadioButton

- 라디오버튼이란?
 - 여러 버튼으로 그룹을 형성하고, 하나만 선택되는 버튼
 - 다른 버튼이 선택되면 이전에 선택된 버튼은 자동으로 해제됨
- 체크박스와의 차이점
 - 체크 박스는 각 체크박스마다 선택/해제 가능
 - 라디오 버튼은 그룹에 속한 버튼 중 하나만 선택 상태가 됨
- 이미지를 가진 라디오버튼의 생성 및 다루기는 체크박스와 완전히 동일



□ 생성자

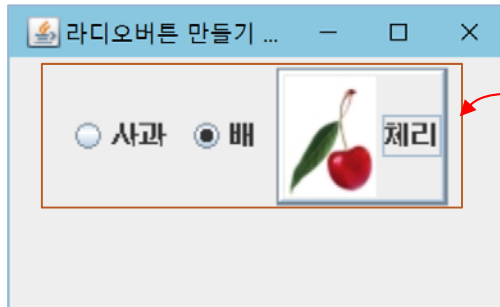
```
JRadioButton() 빈 라디오버튼
JRadioButton(Icon image) 이미지 라디오버튼
JRadioButton(Icon image, boolean selected) 이미지 라디오버튼
JRadioButton(String text) 문자열 라디오버튼
JRadioButton(String text, boolean selected) 문자열 라디오버튼
JRadioButton(String text, Icon image) 문자열과 이미지를 가진 라디오버튼
JRadioButton(String text, Icon image, boolean selected) 문자열과 이미지를 가진 라디오버튼
• selected: true이면 선택 상태로 초기화. 디폴트는 해제 상태
```

라디오버튼 생성 과정

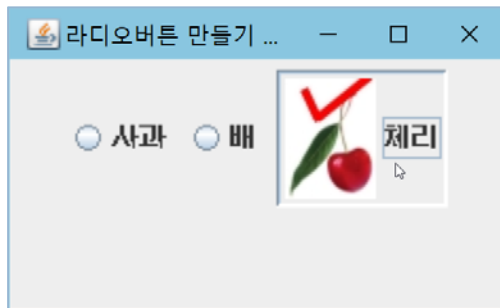
19

1. 버튼 그룹 객체 생성 → `ButtonGroup group = new ButtonGroup();`
2. 라디오버튼 컴포넌트 생성 → `JRadioButton apple= new JRadioButton("사과");`
`JRadioButton pear= new JRadioButton("배");`
`JRadioButton cherry= new JRadioButton("체리");`
3. 라디오 버튼을 버튼 그룹에 삽입 → `group.add(apple);`
`group.add(pear);`
`group.add(cherry);`
4. 라디오 버튼을 컨테이너에 삽입 → `container.add(apple);`
`container.add(pear);`
`container.add(cherry);`

예제 11-6 : 라디오버튼 생성 예



초기 상태(배가 선택된 상태)



체리가 선택된 상태

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

```
public class RadioButtonEx extends JFrame {
```

```
    public RadioButtonEx() {
        setTitle("라디오버튼 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
```

```
        ImageIcon cherryIcon = new ImageIcon("images/cherry.jpg");
        ImageIcon selectedCherryIcon =
            new ImageIcon("images/selectedCherry.jpg");
```

```
        ButtonGroup g = new ButtonGroup();
```

```
        JRadioButton apple = new JRadioButton("사과");
```

```
        JRadioButton pear = new JRadioButton("배", true);
```

```
        JRadioButton cherry = new JRadioButton("체리", cherryIcon);
```

```
        cherry.setBorderPainted(true);
```

```
        cherry.setSelectedIcon(selectedCherryIcon);
```

```
        g.add(apple);
```

```
        g.add(pear);
```

```
        g.add(cherry);
```

```
        c.add(apple);
```

```
        c.add(pear);
```

```
        c.add(cherry);
```

```
        setSize(250,150);
```

```
        setVisible(true);
```

```
    }
```

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

```
        new RadioButtonEx();
```

```
    }
```

```
}
```


예제 11-7 : JRadioButton과 Item 이벤트를 이용하여 과일 사진 보여주기

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

public class RadioButtonItemEventEx extends JFrame {
    private JRadioButton [] radio = new JRadioButton [3];
    private String [] text = {"사과", "배", "체리"};
    private ImageIcon [] image = {
        new ImageIcon("images/apple.jpg"),
        new ImageIcon("images/pear.jpg"),
        new ImageIcon("images/cherry.jpg")};
    private JLabel imageLabel = new JLabel();

    public RadioButtonItemEventEx() {
        setTitle("라디오버튼 Item Event 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new BorderLayout());
        JPanel radioPanel = new JPanel();
        radioPanel.setBackground(Color.GRAY);
        ButtonGroup g = new ButtonGroup();

        for(int i=0; i<radio.length; i++) {
            radio[i] = new JRadioButton(text[i]);
            g.add(radio[i]);
            radioPanel.add(radio[i]);
            radio[i].addItemListener(new MyItemListener());
        }
        radio[2].setSelected(true);
        c.add(radioPanel, BorderLayout.NORTH);
        c.add(imageLabel, BorderLayout.CENTER);
        imageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        setSize(250,200);
        setVisible(true);
    }
}
```

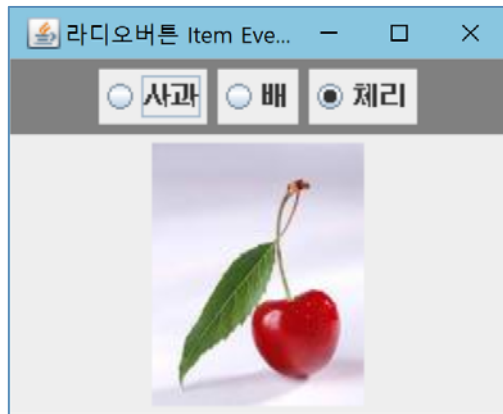
```
class MyItemListener implements ItemListener {
    public void itemStateChanged(ItemEvent e) {
        if(e.getStateChange() == ItemEvent.DESELECTED)
            return;
        if(radio[0].isSelected())
            imageLabel.setIcon(image[0]);
        else if(radio[1].isSelected())
            imageLabel.setIcon(image[1]);
        else
            imageLabel.setIcon(image[2]);
    }
}

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

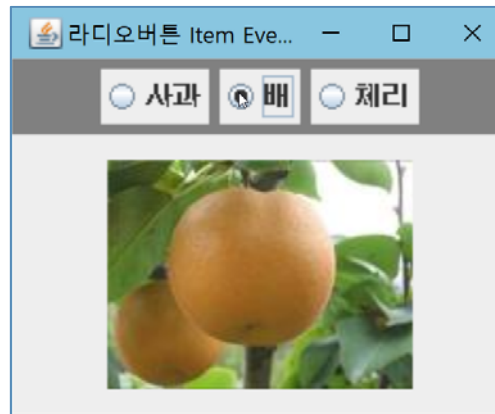
setSelected(true) 호출로 인해 Item 이벤트가 발생하며 해당하는 이미지 출력됨

예제 11-7 실행

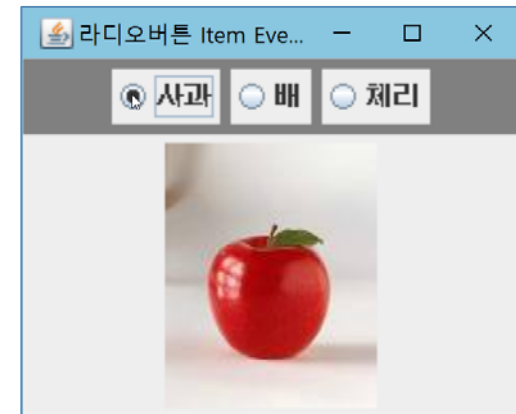
22



초기화면



"배"를 선택한 경우



"사과"를 선택한 경우

JTextField, 텍스트필드 컴포넌트

23

□ JTextField

▣ 텍스트필드란?

- 한 줄 짜리 텍스트(문자열) 입력 창을 구현한 컴포넌트
- 텍스트 입력 도중 <Enter>키가 입력되면 Action 이벤트 발생
- 입력 가능한 문자 개수와 입력 창의 크기는 서로 다름

□ 생성자

`JTextField()` 빈 텍스트필드

`JTextField(int cols)` 입력 창의 열의 개수가 cols개인 텍스트필드

`JTextField(String text)` text 문자열로 초기화된 텍스트필드

`JTextField(String text, int cols)` 입력 창의 열의 개수는 cols개이고 text 문자열로 초기화된 텍스트필드

예제 11-8 : 간단한 텍스트 필드 만들기

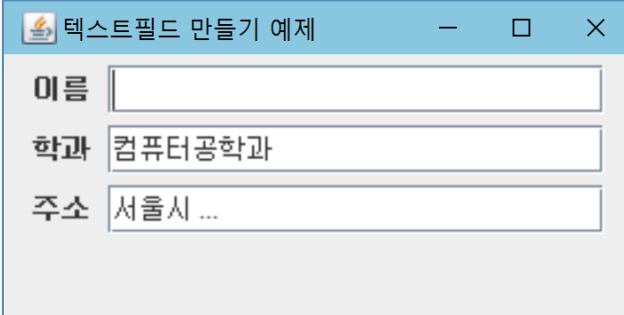
24

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

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

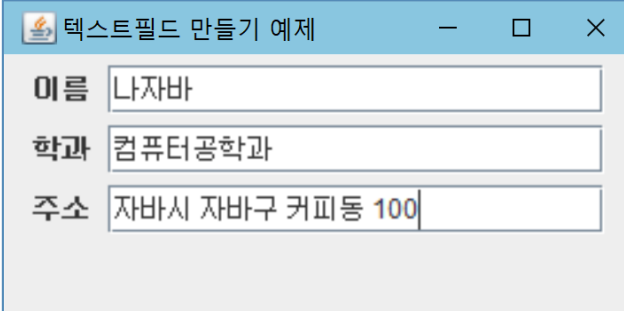
        c.add(new JLabel("이름 "));
        c.add(new JTextField(20));
        c.add(new JLabel("학과 "));
        c.add(new JTextField("컴퓨터공학과 ", 20));
        c.add(new JLabel("주소 "));
        c.add(new JTextField("서울시 ...", 20));

        setSize(300,150);
        setVisible(true);
    }
    public static void main(String [] args) {
        new TextFieldEx();
    }
}
```



The image shows a Java Swing window titled "텍스트필드 만들기 예제". It contains three labels and text fields arranged vertically. The first label is "이름" (Name) followed by an empty text field. The second label is "학과" (Department) followed by a text field containing "컴퓨터공학과" (Computer Engineering). The third label is "주소" (Address) followed by a text field containing "서울시 ..." (Seoul ...).

초기화면



The image shows the same Java Swing window after user input. The "이름" (Name) text field now contains "나자바" (Nazaba). The "학과" (Department) text field still contains "컴퓨터공학과" (Computer Engineering). The "주소" (Address) text field now contains "자바시 자바구 커피동 100" (Java City Java-gu Coffee-dong 100).

사용자가 입력한 경우

TextField의 주요 메소드

25

- 문자열 편집 불가능하게 하기
 - ▣ `TextField.setEditable(false);`
- 입력 창에 문자열 출력
 - ▣ `TextField.setText("hello");`
- 문자열의 폰트 지정
 - ▣ `TextField.setFont(new Font("고딕체", Font.ITALIC, 20);`

TextArea, 텍스트영역 컴포넌트

26

□ JTextArea

▣ 텍스트영역이란?

- 여러 줄을 입력할 수 있는 텍스트 입력 창
- JScrollPane 컴포넌트에 삽입하면 스크롤바 지원됨

□ 생성자

`JTextArea()` 빈 텍스트영역

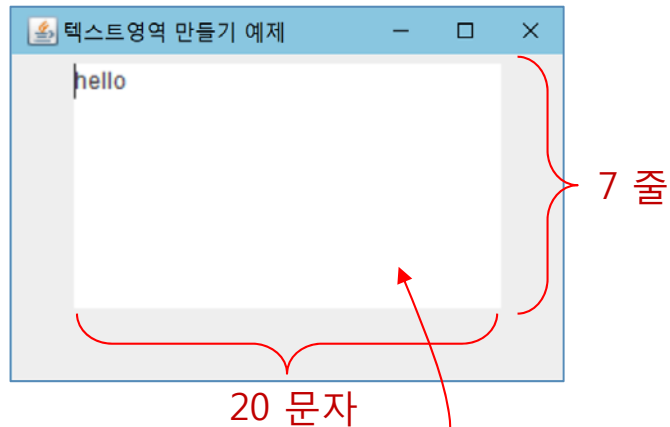
`JTextArea(int rows, int cols)` 입력 창이 rows × cols개의 문자 크기인 텍스트영역

`JTextArea(String text)` text 문자열로 초기화된 텍스트영역

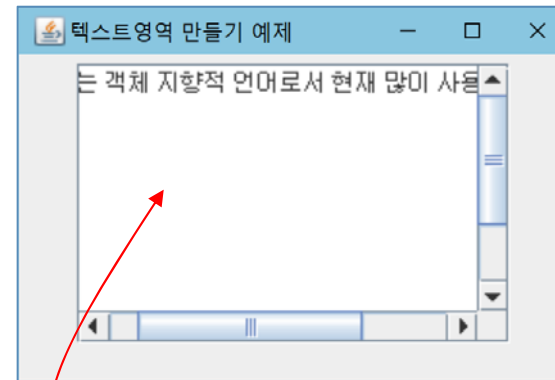
`JTextArea(String text, int rows, int cols)` 입력 창이 rows × cols개의 문자 크기이며
text 문자열로 초기화된 텍스트영역

스크롤 가능한 텍스트영역 만들기

27

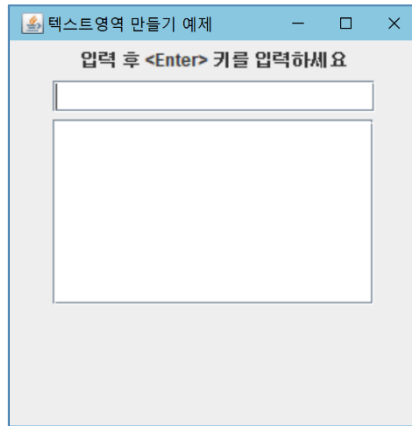


`new JTextArea("hello", 7, 20);`

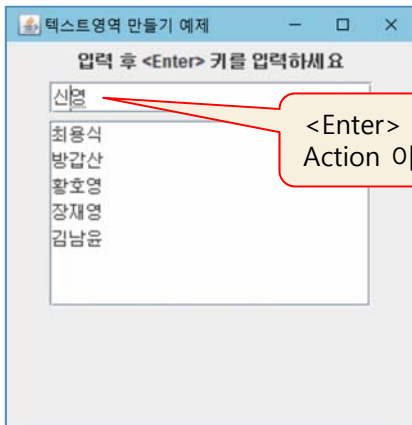


`new JScrollPane(new JTextArea("hello", 7, 20));`

예제 11-9 : JTextArea 컴포넌트 생성



초기화면



<Enter> 키를 입력하면
Action 이벤트 발생

텍스트필드에 입력하고
<Enter> 키를 누르면 텍스트필드에
입력한 문자열을 텍스트 영역창에 추가

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

public class TextAreaEx extends JFrame {
    private JTextField tf = new JTextField(20);
    private JTextArea ta = new JTextArea(7, 20);

    public TextAreaEx() {
        setTitle("텍스트영역 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        c.add(new JLabel("입력 후 <Enter> 키를 입력하세요"));
        c.add(tf);
        c.add(new JScrollPane(ta));

        tf.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JTextField t = (JTextField)e.getSource();
                ta.append(t.getText() + "\n");
                t.setText("");
            }
        });
        setSize(300,300);
        setVisible(true);
    }

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

<Enter>키가 입력되면
tf에 입력된 문자열을
ta 끝에 추가

JList<E>, 리스트 컴포넌트

29

□ JList<E>

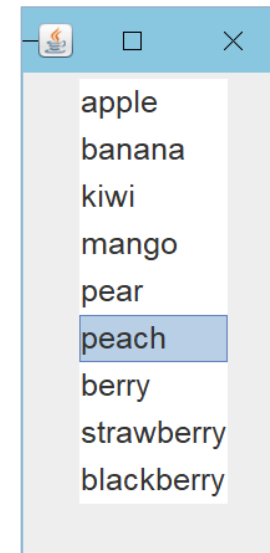
▣ 리스트 컴포넌트란?

- 여러 개의 아이템을 리스트 형식으로 보여주고 선택하는 컴포넌트
- JComboBox<E>와 기본적으로 같은 기능
- JScrollPane에 JList<E>를 삽입하여 스크롤 가능

▣ JList<E>

- JDK7부터 제네릭 리스트로 바뀜
- <E>에 지정된 타입의 객체만 저장하는 리스트

JList<String>의 문자열 리스트



□ 생성자

`JList<E>()` 빈 리스트

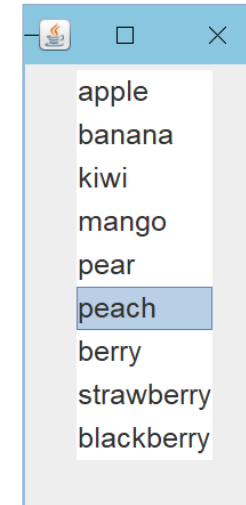
`JList<E>(Vector listData)` 벡터로부터 아이템을 공급받는 리스트

`JList<E>(Object [] listData)` 배열로부터 아이템을 공급받는 리스트

리스트를 생성하는 방법

1. 객체 배열로 아이템 제공

```
String [] fruits= {"apple", "banana", "kiwi", "mango", "pear", "peach",  
                  "berry", "strawberry", "blackberry"};  
JList<String> strList = new JList<String>(fruits);
```

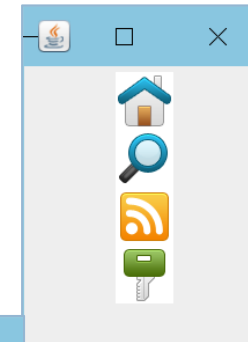


2. Vector로 아이템 제공

```
Vector v = new Vector();  
v.add("apple");  
v.add("banana");  
v.add("kiwi");  
JList<String> vList = new JList<String>(v);
```

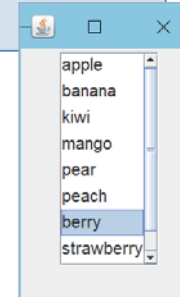
3. 빈 JList 컴포넌트를 생성하고 setListData()로 아이템 제공

```
ImageIcon [] images = {new ImageIcon("images/icon1.png"),  
                       new ImageIcon("images/icon2.png"),  
                       new ImageIcon("images/icon3.png"),  
                       new ImageIcon("images/icon4.png")  
};  
JList<ImageIcon> imageList = new JList<ImageIcon>();  
imageList.setListData(images);
```



4. 스크롤 지원

```
JList<String> scrollList = new JList<String>(fruits);  
new JScrollPane(scrollList);
```



예제 11-10 : 리스트 만들기



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

public class ListEx extends JFrame {
    private String [] fruits= {"apple", "banana", "kiwi", "mango", "pear",
                               "peach", "berry", "strawberry", "blackberry"};
    private ImageIcon [] images = {
        new ImageIcon("images/icon1.png"),
        new ImageIcon("images/icon2.png"),
        new ImageIcon("images/icon3.png"),
        new ImageIcon("images/icon4.png") };

    public ListEx() {
        setTitle("리스트 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        JList<String> strList = new JList<String>(fruits);
        c.add(strList);

        JList<ImageIcon> imageList = new JList<ImageIcon>();
        imageList.setListData(images);
        c.add(imageList);

        JList<String> scrollList = new JList<String>(fruits);
        c.add(new JScrollPane(scrollList));

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

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

리스트의 아이템 변경

32

□ JList<E>의 특징

- ▣ JList<E>(Vector listData)나 JList<E>(Object [] listData)로 리스트가 생성되고 나면 벡터나 배열을 수정해도 리스트 수정 안됨

□ 리스트를 수정하는 간단한 방법

- ▣ JList<E>의 setListData()를 호출

- 리스트에 수정된 벡터나 배열을 새로 달아주는 방법

```
Vector<String> v = new Vector<String>();  
v.add("황기태");  
v.add("이재문");  
JList<String> nameList = new JList<String>(v);
```

```
// 벡터 v를 수정하고, 벡터 v를 리스트에 다시 달기  
v.add("김남윤");  
nameList.setListData(v);
```


예제 11-11 : 리스트의 아이템 변경

33

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

public class ListChangeEx extends JFrame {
    private JTextField tf = new JTextField(10);
    private Vector<String> v = new Vector<String>();
    private JList<String> nameList = new JList<String>(v);

    public ListChangeEx() {
        setTitle("리스트 변경 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

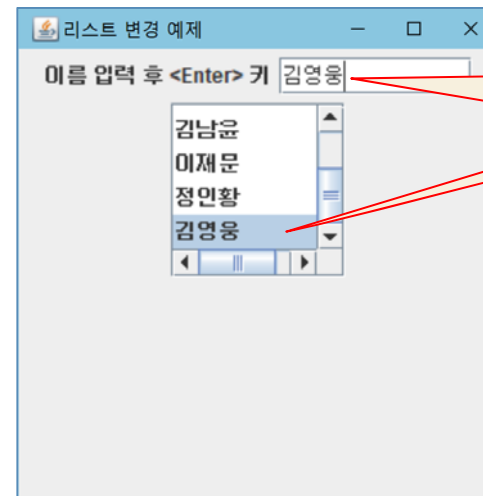
        c.add(new JLabel("이름 입력 후 <Enter> 키"));
        c.add(tf);

        v.add("황기태");
        v.add("이재문");
        nameList.setVisibleRowCount(5);
        nameList.setFixedCellWidth(100);
        c.add(new JScrollPane(nameList));

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

```
// JTextField에 ActionListener 등록. <Enter> 키 처리
tf.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JTextField t = (JTextField)e.getSource();
        v.add(t.getText());
        t.setText("");
        nameList.setListData(v);
    }
});

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



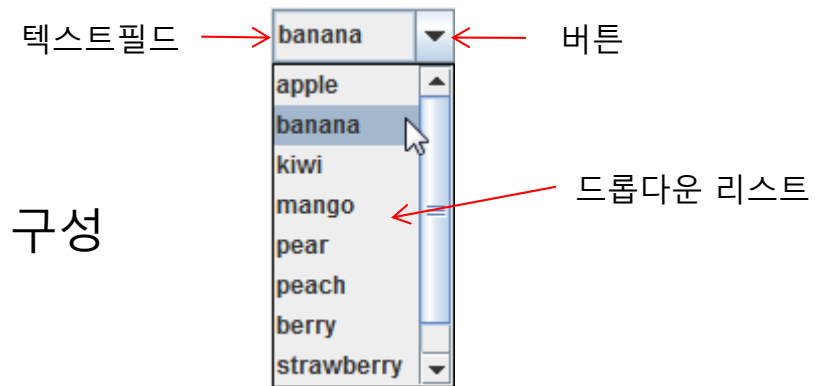
JComboBox<E>, 콤보박스 컴포넌트

34

□ JComboBox<E>

▣ 콤보박스란?

- 텍스트 필드와 버튼,
그리고 드롭다운 리스트로 구성



JComboBox<String> 콤보박스

□ 생성자

`JComboBox<E>()` 빈 콤보박스

`JComboBox<E>(Vector<E> listData)` 벡터로부터 아이템을 공급받는 콤보박스

`JComboBox<E>(Object[] listData)` 배열로부터 아이템을 공급받는 콤보박스

예제 11-12 : 콤보박스 만들기

35

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

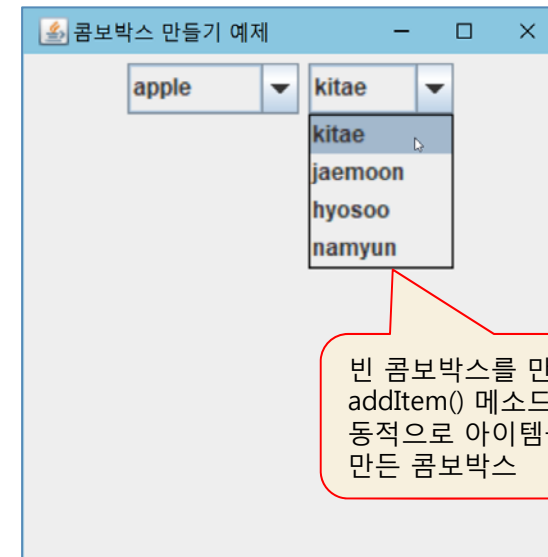
public class ComboBoxEx extends JFrame {
    private String [] fruits = {"apple", "banana", "kiwi", "mango", "pear",
                                "peach", "berry", "strawberry", "blackberry"};
    private String [] names = {"kitae", "jaemoon", "hyosoo", "namyun"};
    public ComboBoxEx() {
        setTitle("콤보박스 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());

        JComboBox<String> strCombo = new JComboBox<String>(fruits);
        c.add(strCombo);

        JComboBox<String> nameCombo = new JComboBox<String>();
        for(int i=0; i<names.length; i++)
            nameCombo.addItem(names[i]);
        c.add(nameCombo);

        setSize(300,300);
        setVisible(true);
    }
    public static void main(String [] args) {
        new ComboBoxEx();
    }
}
```

addItem() 메소드를 호출하여
아이템 동적 삽입



JComboBox<E>와 Action 이벤트

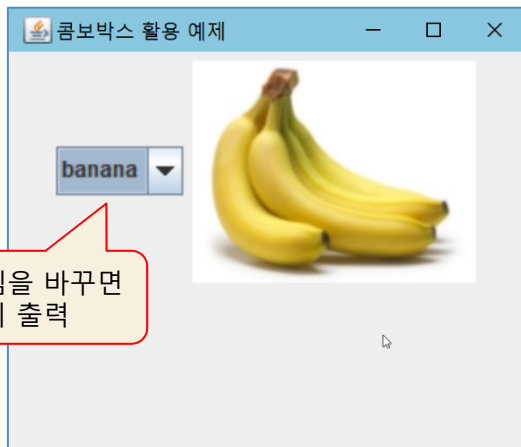
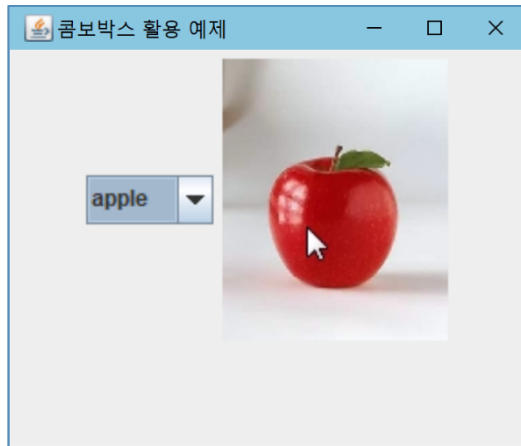
36

- 콤보박스의 아이템 선택시 Action 이벤트 발생
- 현재 선택된 아이템 알아내기
 - ▣ JComboBox<E>의 다음 메소드 활용

int *getSelectedIndex()* 선택 상태인 아이템의 인덱스 번호를 리턴한다.

Object *getSelectedItem()* 선택 상태인 아이템 객체의 레퍼런스를 리턴한다.

예제 11-13 : Action 이벤트를 이용한 콤보박스 활용 예



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

```
public class ComboActionEx extends JFrame {
    private String [] fruits = {"apple", "banana", "kiwi", "mango"};
    private ImageIcon [] images = { new ImageIcon("images/apple.jpg"),
                                     new ImageIcon("images/banana.jpg"),
                                     new ImageIcon("images/kiwi.jpg"),
                                     new ImageIcon("images/mango.jpg")};

    private JLabel imgLabel = new JLabel(images[0]);
    private JComboBox<String> strCombo = new JComboBox<String>(fruits);

    public ComboActionEx() {
        setTitle("콤보박스 활용 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        c.add(strCombo);
        c.add(imgLabel);

        strCombo.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JComboBox<String> cb = (JComboBox<String>)e.getSource();
                int index = cb.getSelectedIndex();
                imgLabel.setIcon(images[index]);
            }
        });
        setSize(300,250);
        setVisible(true);
    }

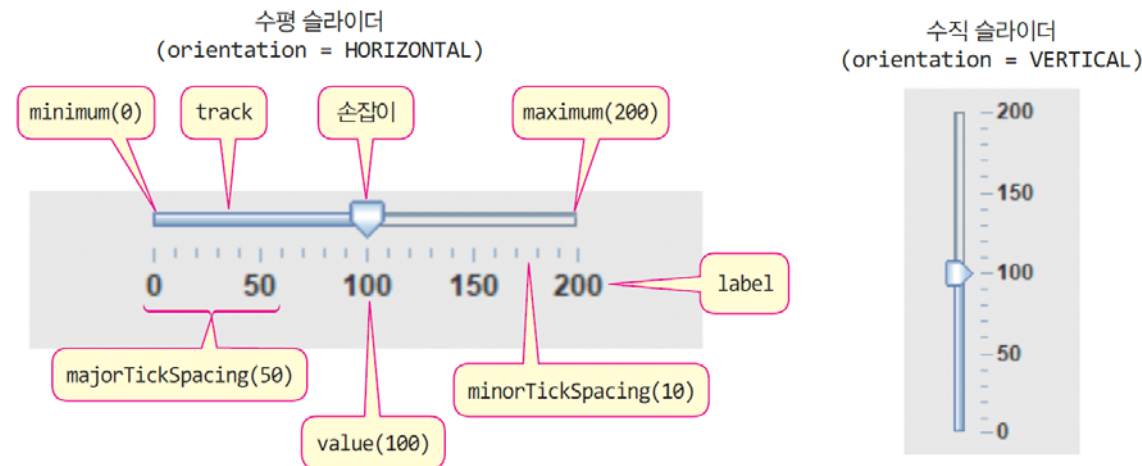
    public static void main(String [] args) {
        new ComboActionEx();
    }
}
```

JSlider, 슬라이더

38

□ JSlider

- ▣ 슬라이더란? 마우스로 움직이면서 값을 선택하는 컴포넌트



□ 슬라이더 생성

`JSlider()` 디폴트 슬라이더 생성

`JSlider(int orientation)` `orientation` 방향의 슬라이더 생성

`JSlider(int min, int max, int val)` 최소(`min`), 최대(`max`), 초깃값(`val`)을 가진 슬라이더 생성

`JSlider(int orientation, int min, int max, int val)`

- `orientation`은 `JSlider.HORIZONTAL`과 `JSlider.VERTICAL` 중 하나이며 각각 수평 슬라이더와 수직 슬라이더를 의미한다.

- `min`, `max`, `val` 값은 각각 `minimum`, `maximum`, `value`의 초깃값이다.

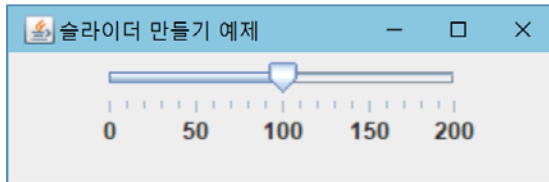
슬라이더의 모양 제어

39

- 슬라이더 방향 설정
 - ▣ void setOrientation(int orientation)
 - orientation : JSlider.HORIZONTAL, JSlider.VERTICAL
- 최대 최소 값 설정
 - ▣ void setMaximum(int max)
 - ▣ void setMinimum(int min)
- label 보이기/감추기
 - ▣ void setPaintLabels(boolean b)
 - b가 true이면 label 출력
- tick 보이기/감추기
 - ▣ void setPaintTicks(boolean b)
 - b가 true이면 눈금 출력
- track 보이기/감추기
 - ▣ void setPaintTrack(boolean b)
 - b가 true이면 track 출력
- 큰 눈금 간격 지정
 - ▣ void setMajorTickSpacing(int space)
- 작은 눈금 간격 지정
 - ▣ void setMinorTickSpacing(int space)
- 슬라이더 값 제어
 - ▣ void setValue(int n)
 - n이 슬라이더의 값이 되며 이에 따라 슬라이더의 손잡이 위치가 변경된다.

예제 11-14 : JSlider로 슬라이더 컴포넌트를 만들고 모양을 제어하는 예

40



0~200 사이의 슬라이더

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

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

        JSlider slider = new JSlider(JSlider.HORIZONTAL, 0, 200, 100);
        slider.setPaintLabels(true);
        slider.setPaintTicks(true);
        slider.setPaintTrack(true);
        slider.setMajorTickSpacing(50);
        slider.setMinorTickSpacing(10);

        c.add(slider);
        setSize(300,100);
        setVisible(true);
    }
    public static void main(String [] args) {
        new SliderEx();
    }
}
```


JSlider와 Change 이벤트

41

□ Change 이벤트

▣ JSlider의 값(value)이 바뀌면 발생

- 사용자가 슬라이더의 손잡이를 움직이는 경우
- JSlider의 setValue(int n)를 호출하여 값(value 필드)이 바뀌는 경우

□ ChangeListener의 메소드

```
void stateChanged(ChangeEvent e)
```

컴포넌트의 상태가 변할 때 호출되며 ChangeEvent 객체를 인자로 전달받는다.

예제 11-15 : JSlider와 Change이벤트를 활용한 색깔 다루기

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

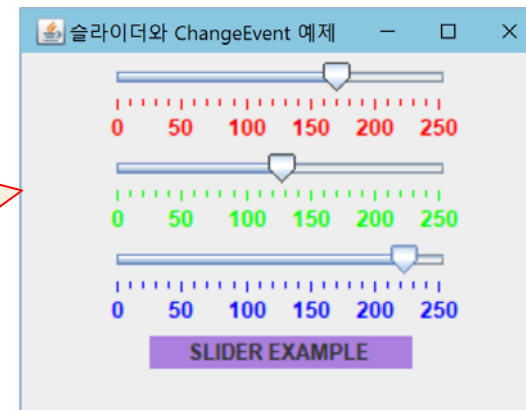
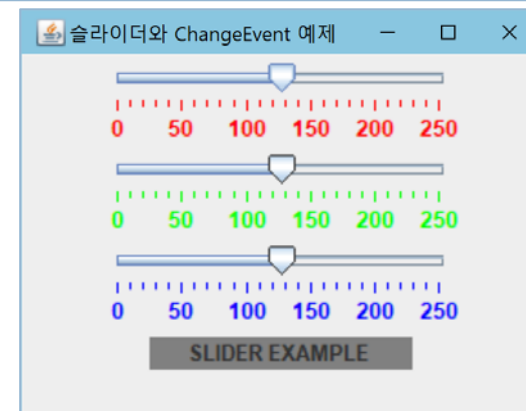
public class SliderChangeEx extends JFrame {
    private JLabel colorLabel;
    private JSlider [] sl = new JSlider [3];
    public SliderChangeEx() {
        setTitle("슬라이더와 ChangeEvent 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        colorLabel = new JLabel(" SLIDER EXAMPLE ");
        for(int i=0; i<sl.length; i++) {
            sl[i] = new JSlider(JSlider.HORIZONTAL, 0, 255, 128);
            sl[i].setPaintLabels(true);
            sl[i].setPaintTicks(true);
            sl[i].setPaintTrack(true);
            sl[i].setMajorTickSpacing(50);
            sl[i].setMinorTickSpacing(10);
            sl[i].addChangeListener(new MyChangeListener());
            c.add(sl[i]);
        }
        sl[0].setForeground(Color.RED);
        sl[1].setForeground(Color.GREEN);
        sl[2].setForeground(Color.BLUE);

        int r = sl[0].getValue();
        int g = sl[1].getValue();
        int b = sl[2].getValue();

        colorLabel.setOpaque(true);
        colorLabel.setBackground(new Color(r,g,b));
        c.add(colorLabel);
        setSize(300,230);
        setVisible(true);
    }
}
```

```
class MyChangeListener implements ChangeListener {
    public void stateChanged(ChangeEvent e) {
        int r = sl[0].getValue();
        int g = sl[1].getValue();
        int b = sl[2].getValue();
        colorLabel.setBackground(new Color(r,g,b));
    }
}

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



슬라이더의 손잡이를 움직여 색깔 만들기