

Java Core

제 16 강

AWT & Swing

1. Frame
2. 버튼, 레이아웃
3. Choice, Menu
4. 각종 리스너
5. Awt 실전예제

자바의 GUI(Graphical User Interface)

- GUI 목적
 - 그래픽 이용, 사용자에게 이해하기 쉬운 모양으로 정보 제공
 - 사용자는 마우스나 키보드를 이용하여 쉽게 입력
- 자바 GUI 특징
 - 뛰어난 GUI 컴포넌트 제공(스윙은 제외)
 - 쉬운 GUI 프로그래밍
- 자바의 GUI 프로그래밍 방법
 - GUI 컴포넌트와 그래픽 이용
 - AWT 패키지와 Swing 패키지에 제공되는 메카니즘 이용
 - AWT - java.awt 패키지
 - Swing - javax.swing 패키지

AWT와 Swing 패키지

- AWT(Abstract Windowing Toolkit)
 - 자바가 처음 나왔을 때 함께 배포된 GUI 라이브러리
 - java.awt 패키지
 - native(운영체제)와 응용프로그램 사이의 연결 라이브러리
 - 중량 컴포넌트(Heavy weight components)
 - AWT 컴포넌트는 native(peer)에 의존적임
 - OS의 도움을 받아야 화면에 출력되며 동작하는 컴포넌트. 운영체제에 많은 부담. 오히려 처리 속도는 빠름
- Swing(스윙)
 - AWT 기술을 기반으로 작성된 자바 라이브러리
 - 모든 AWT 기능 + 추가된 풍부하고 화려한 고급 컴포넌트
 - AWT 컴포넌트에 J자가 덧붙여진 이름의 클래스
 - 그 외 J자로 시작하는 클래스
 - 순수 자바 언어로 구현, JDK 1.1 부터 - javax.swing 패키지
 - Swing 컴포넌트는 native(peer 혹은 운영체제)에 의존하지 않음
 - 경량 컴포넌트(Light weight components)

스윙 컴포넌트 예시



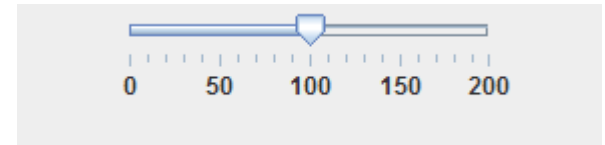
JButton



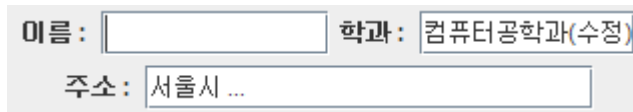
JCheckBox



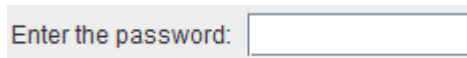
JRadioButton



JSlider



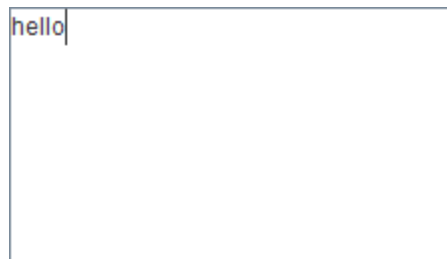
JTextField



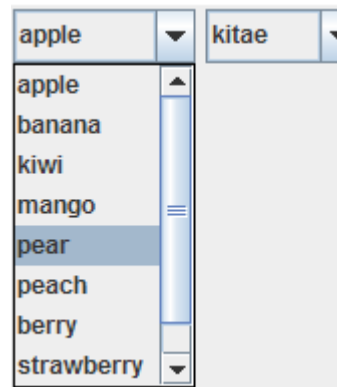
JPasswordField



JSpinner



JTextArea

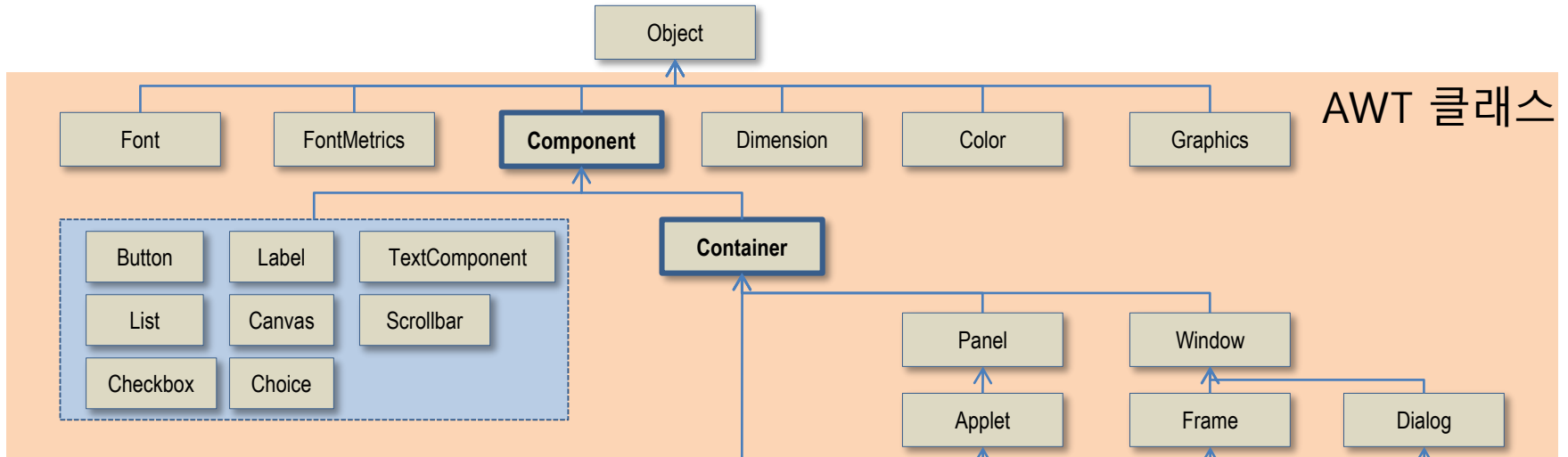


JComboBox

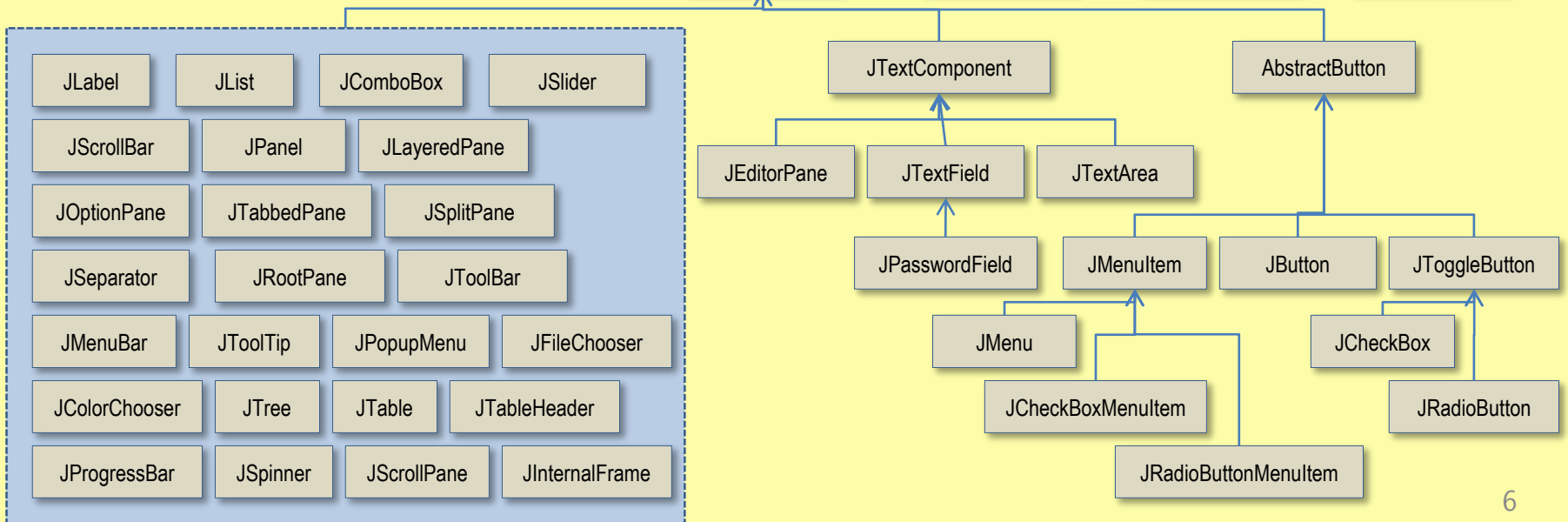


JList

GUI 라이브러리 계층 구조



Swing 클래스



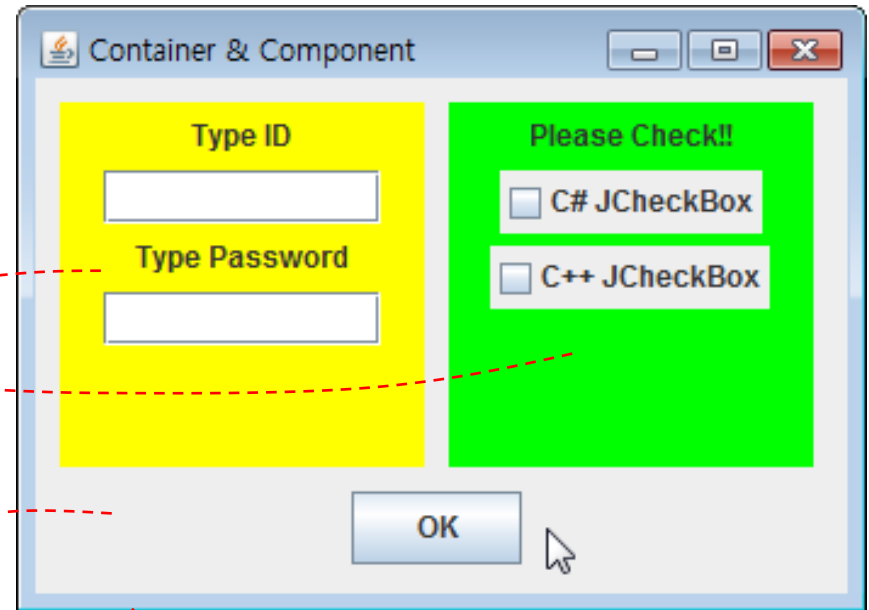
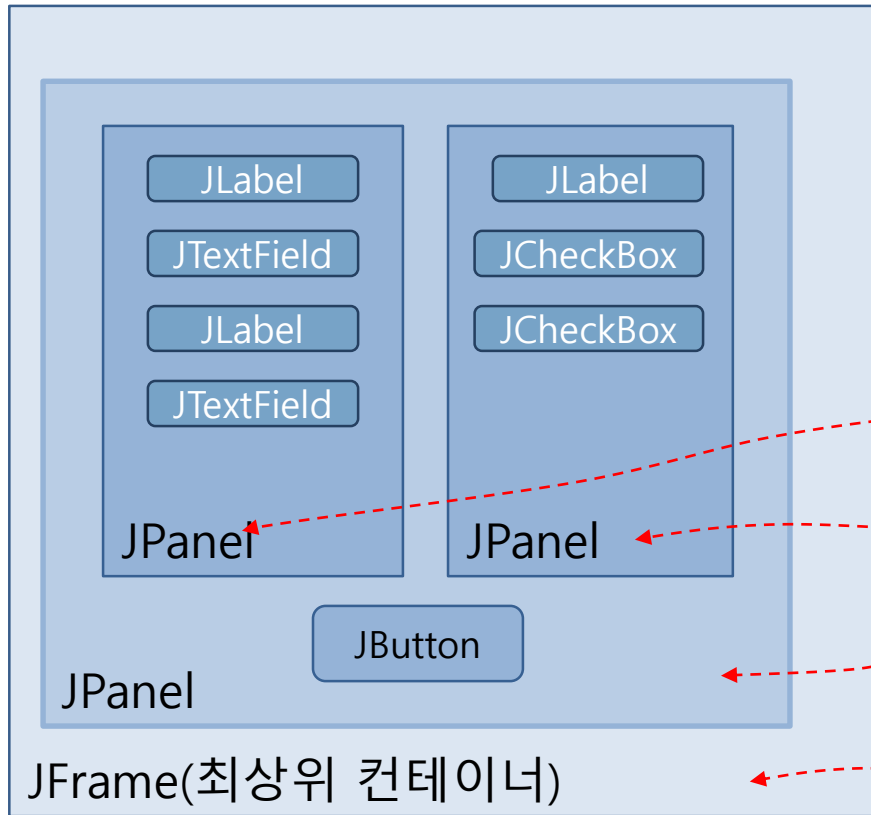
Swing 클래스의 특징

- 클래스 이름이 J 자로 시작
- 화려하고 다양한 컴포넌트로 쉽게 GUI 프로그래밍
- 스윙 컴포넌트는 2 가지 유형
 - JComponent는 상속받는 클래스
 - 대부분의 스윙 컴포넌트
 - AWT의 Container를 상속받는 몇 개의 클래스
 - JApplet, JDialog, JFrame 등
- JComponent
 - 매우 중요한 추상 클래스
 - 스윙 컴포넌트의 공통적인 속성 구현
 - ~~new JComponent()~~ 인스턴스를 생성할 수 없음
 - AWT의 Component를 상속받음

컨테이너와 컴포넌트

- 컨테이너
 - 다른 컴포넌트를 포함할 수 있는 GUI 컴포넌트
 - java.awt.Container를 상속받음
 - 다른 컨테이너에 포함될 수 있음
 - AWT 컨테이너
 - Panel, Frame, Applet, Dialog, Window
 - Swing 컨테이너
 - JPanel JFrame, JApplet, JDialog, JWindow
- 최상위 컨테이너
 - 다른 컨테이너에 속하지 않고 독립적으로 존재 가능한 컨테이너
 - 스스로 화면에 자신을 출력하는 컨테이너
 - JFrame, JDialog, JApplet
 - 모든 컴포넌트는 컨테이너에 포함되어야 화면에 출력 가능
- 컴포넌트
 - 컨테이너에 포함되어야 화면에 출력될 수 있는 GUI 객체
 - 모든 GUI 컴포넌트의 최상위 클래스
 - java.awt.Component
 - 스윙 컴포넌트의 최상위 클래스
 - javax.swing.JComponent

컨테이너와 컴포넌트의 포함관계



스윙 GUI 프로그램

스윙의 컨테이너와 컴포넌트의 포함 관계

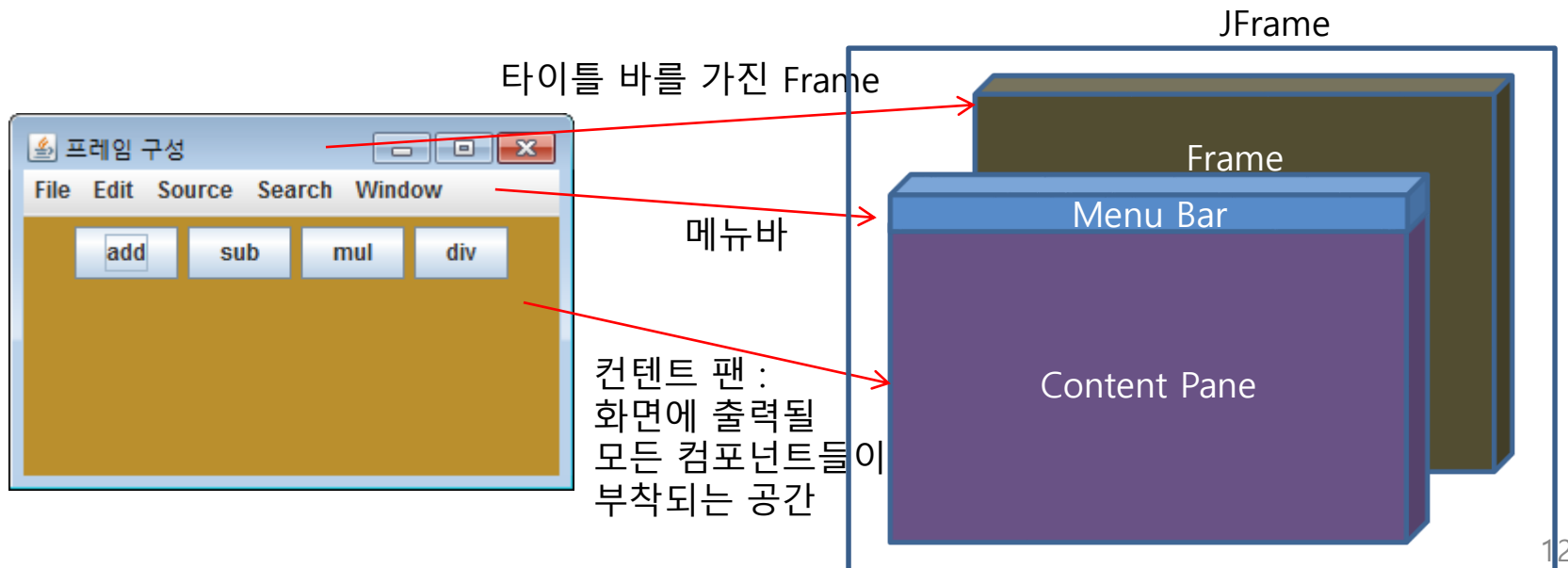
1. Frame

스윙 GUI 프로그램 만들기

1. 프레임 만들기
 2. 프레임에 스윙 컴포넌트 붙이기
 3. main() 메소드 작성
- 스윙 프로그램을 작성하기 위한 import문
 - `import java.awt.*;` // 그래픽 처리를 위한 클래스들의 경로명
 - `import java.awt.event.*;` // AWT 이벤트 사용을 위한 경로명
 - `import javax.swing.*;` // 스윙 컴포넌트 클래스들의 경로명
 - `import javax.swing.event.*;` // 스윙 이벤트를 위한 경로명

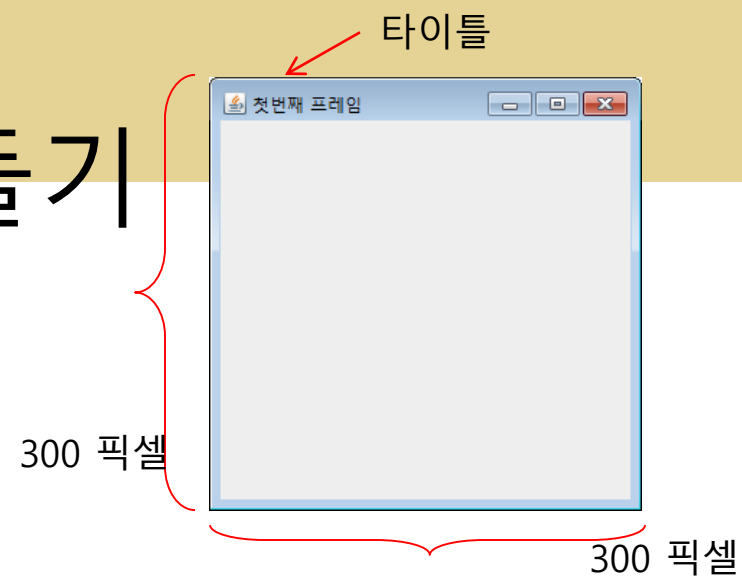
스윙 프레임

- 모든 스윙 컴포넌트를 담는 최상위 GUI 컨테이너
 - JFrame을 상속받아 구현
 - 컴포넌트가 화면에 보이려면 스윙 프레임에 부착되어야 함
 - 프레임을 닫으면 프레임 내의 모든 컴포넌트가 보이지 않게 됨
- 스윙 프레임(JFrame) 기본 구성
 - 프레임 - 스윙 프로그램의 기본 틀
 - 메뉴바 - 메뉴들이 부착되는 공간
 - 콘텐츠 팬 - GUI 컴포넌트들이 부착되는 공간



프레임 만들기

• 두 가지 방법



- main() 메소드에서 JFrame 객체를 생성
- 확장성, 융통성 결여

```
import javax.swing.*;

public class MyApp {
    public static void main(String [] args) {
        JFrame f = new JFrame();
        f.setTitle("첫번째 프레임");
        f.setSize(300,300);
        f.setVisible(true);
    }
}
```

방법 1. main() 메소드에서 JFrame 객체 생성

- JFrame 을 상속받은 프레임 클래스 이용
- main() 은 단순히 프레임 객체를 생성하는 역할

```
import javax.swing.*;

public class MyFrame extends JFrame {
    MyFrame() {
        setTitle("첫번째 프레임");
        setSize(300,300);
        setVisible(true);
    }

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

추천

방법 2. JFrame 을 상속받은 프레임 클래스 이용

main()의 우

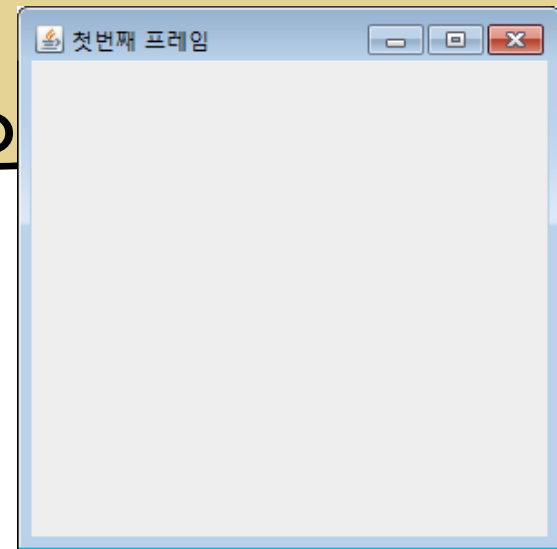
추천

```
import javax.swing.*;

public class MyFrame extends JFrame {
    MyFrame() {
        setTitle("첫번째 프레임");
        setSize(300,300);
        setVisible(true);
    }

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

main()을 프레임 클래스 내의 멤버로 작성



```
import javax.swing.*;

class MyFrame extends JFrame {
    MyFrame() {
        setTitle("첫번째 프레임");
        setSize(300,300);
        setVisible(true);
    }
}

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

main()을 가진 다른 클래스 MyApp 작성

프레임에 컴포넌트 붙이기

타이틀 – 타이틀 바에 부착

```
// JFrame의 생성자 이용  
JFrame frame = new JFrame("타이틀문자열");  
  
// JFrame의 setTitle() 메소드 호출  
frame.setTitle("타이틀문자열");
```

컨텐츠팬 알아내기

```
JFrame frame = new JFrame();  
  
Container contentPane = frame.getContentPane();
```

스윙 컴포넌트 – 컨텐츠 팬에 부착

컨텐츠팬에 컴포넌트 달기

```
JFrame frame = new JFrame();  
JButton b = new JButton("Click");  
Container c = frame.getContentPane();  
c.add(b);
```

컨텐츠팬 변경

```
JPanel p = new JPanel();  
frame.setContentPane(p);
```

예제: Frame 만들기 예

- Frame_make1.java
 - JFrame 으로 객체를 선언하여 만듦
- Frame_make2.java
 - JFrame 을 상속한 클래스로 만듦

2. 버튼, 레이아웃

Tip. 컨테트팬에 대한 JDK1.5이후의 변경 사항

– JDK 1.5 이전

- 프레임의 컨테트팬을 알아내어 반드시 컨테트팬에 컴포넌트 부착

```
JFrame frame = new JFrame();  
JButton b = new JButton("Click");  
Container c = frame.getContentPane();  
c.add(b); // 버튼 b를 컨테트팬에 부착
```

– JDK 1.5 이후

- 프레임에 컴포넌트를 부착하면 프레임이 대신 컨테트팬에 부착

```
JFrame frame = new JFrame();  
JButton b = new JButton("Click");  
frame.add(b); // 컨테트팬에 대신 버튼 b 부착
```

– 결론

- JDK 1.5 이전처럼 명료하게 컨테트팬에 컴포넌트를 부착하는 것이 바람직함
 - 컨테트팬에 접근하고 다루는 경우가 많기 때문

예제 : 컴포넌트를 부착한 프레임 예

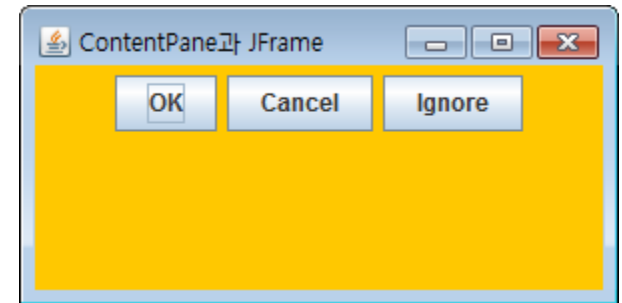
```
import javax.swing.*;
import java.awt.*;

public class Contentpane extends JFrame {
    Contentpane () {
        setTitle("ContentPane과 JFrame");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        Container contentPane = getContentPane();
        contentPane.setBackground(Color.ORANGE);
        contentPane.setLayout(new FlowLayout());
        contentPane.add(new JButton("OK"));
        contentPane.add(new JButton("Cancel"));
        contentPane.add(new JButton("Ignore"));

        setSize(350, 150);
        setVisible(true);
    }

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



스윙 응용프로그램의 종료

- 응용프로그램 내에서 스스로 종료

```
System.exit(0);
```

- 언제 어디서나 무조건 종료
- **프레임 종료버튼(X)이 클릭되면 어떤 일이 일어나는가?**
 - 프레임을 종료하여 프레임 윈도우가 닫힘
 - 프레임이 화면에서 보이지 않게 되고 응용프로그램이 사라짐
 - 프레임이 보이지 않게 되지만 응용프로그램이 종료한 것 아님
 - 키보드나 마우스 입력을 받지 못함
 - 다시 setVisible(true)를 호출하면 보이게 되고 이전 처럼 작동함
- 프레임 종료버튼이 클릭될 때 프레임을 닫고 응용 프로그램이 종료하도록 하는 방법

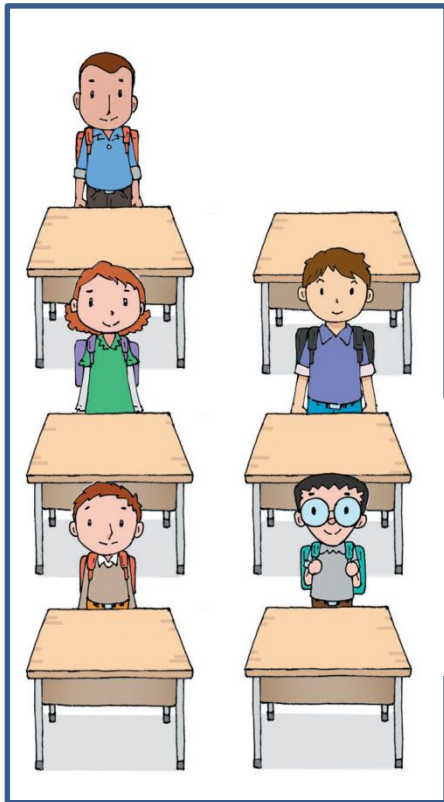
```
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

main() 종료 뒤에도 프레임이 살아 있는 이유?

- 스윙 프로그램이 실행되는 동안 생성되는 스레드
 - 메인 스레드
 - main()을 실행하는 스레드
 - 자바 응용프로그램의 실행을 시작한 스레드
 - 이벤트 처리 스레드
 - 스윙 응용프로그램이 실행될 때 자동으로 실행되는 스레드
 - 이벤트 처리 스레드의 역할
 - 프레임과 버튼 등 GUI 화면 그리기
 - 키나 마우스 입력을 받아 이벤트를 처리할 코드 호출
- 자바 응용프로그램의 종료 조건
 - 실행 중인 사용자 스레드가 하나도 없을 때 종료
- 스윙 프로그램 main() 종료 뒤 프레임이 살아있는 이유
 - 메인 스레드가 종료되어도 이벤트 처리 스레드가 살아 있어 프레임 화면을 그리고 마우스나 키 입력을 받기 때문

컨테이너와 배치 개념

컨테이너(Container)



이쪽으로
가세요.



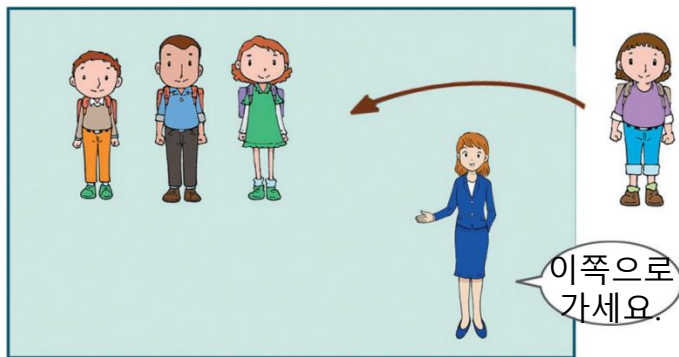
컴포넌트
(Component)

1. 컨테이너마다 하나의 배치관리자가 존재하며, 삽입되는 모든 컴포넌트의 위치와 크기를 결정하고 적절히 배치한다.
2. 컨테이너의 크기가 변하면 내부 컴포넌트들의 위치와 크기를 모두 재조절하고 재배치한다.

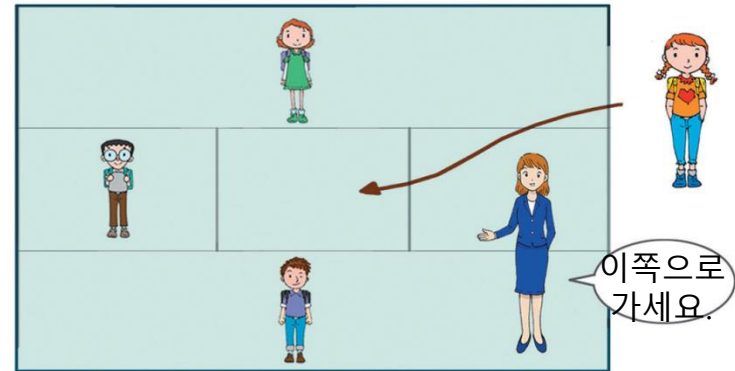
배치 관리자 대표 유형 4 가지

- java.awt 패키지에 구현되어 있음

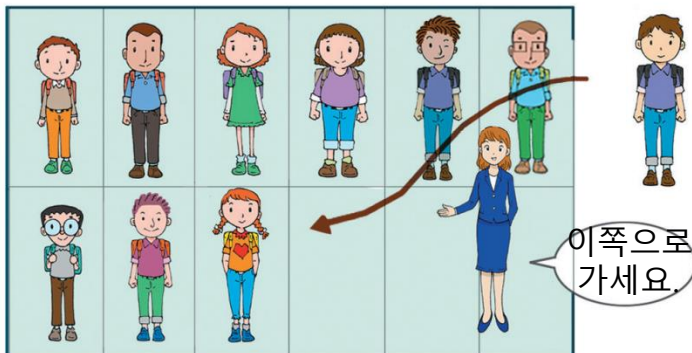
FlowLayout



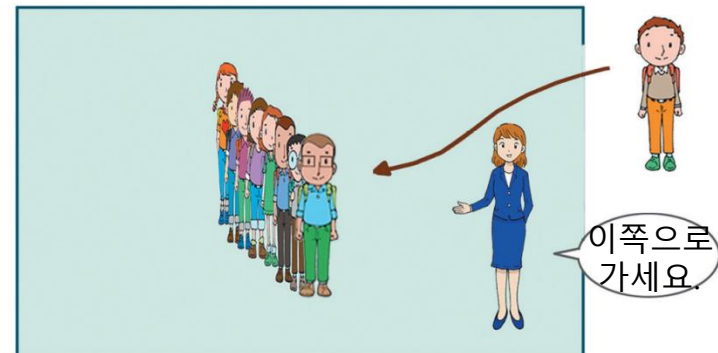
BorderLayout



GridLayout



CardLayout



컨테이너와 배치관리자

- 컨테이너의 디폴트 배치관리자
 - 컨테이너는 생성시 디폴트 배치관리자 설정

| AWT와 스윙 컨테이너 | 디폴트 배치관리자 |
|-----------------|--------------|
| Window, JWindow | BorderLayout |
| Frame, JFrame | BorderLayout |
| Dialog, JDialog | BorderLayout |
| Panel, JPanel | FlowLayout |
| Applet, JApplet | FlowLayout |

- 컨테이너에 새로운 배치관리자 설정
 - Container.setLayout(LayoutManager lm)
 - lm을 새로운 배치관리자로 설정

// JPanel 패널에 BorderLayout 배치관리자를 설정하는 예

```
JPanel p = new JPanel();  
p.setLayout(new BorderLayout());
```

```
JFrame frame = new JFrame();  
Container c = frame.getContentPane(); // 프레임의 콘텐츠팬  
c.setLayout(new FlowLayout()); // 콘텐츠팬에 FlowLayout 설정
```

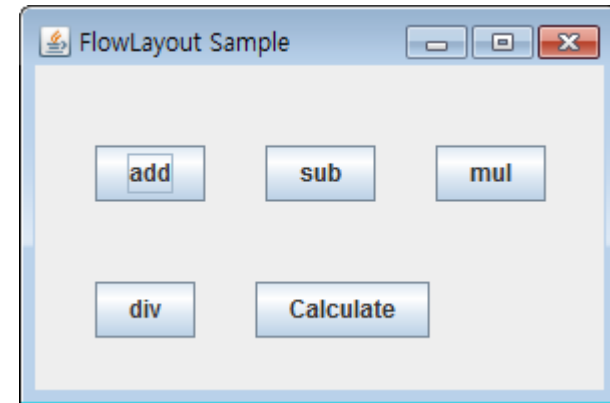
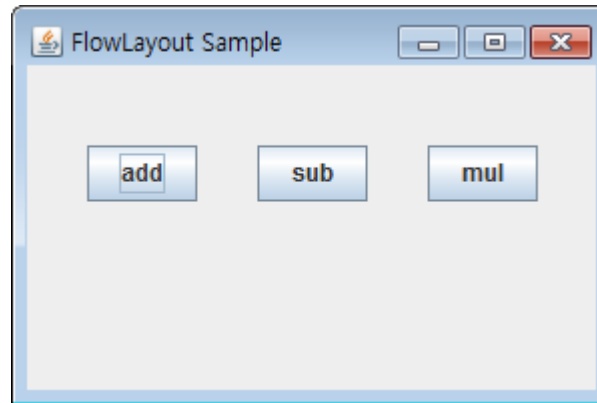
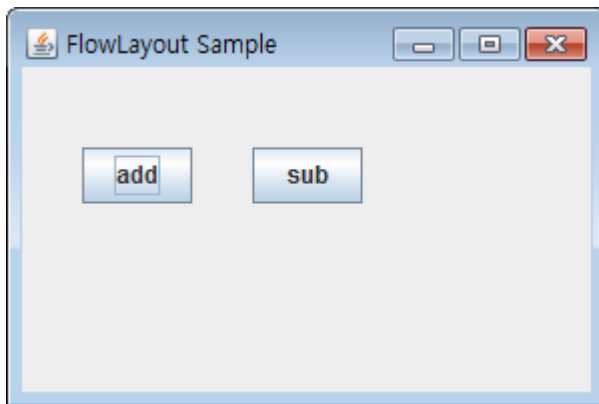
혹은

```
frame.setLayout(new FlowLayout()); // JDK 1.5 이후 버전에서
```


FlowLayout

- 배치방법

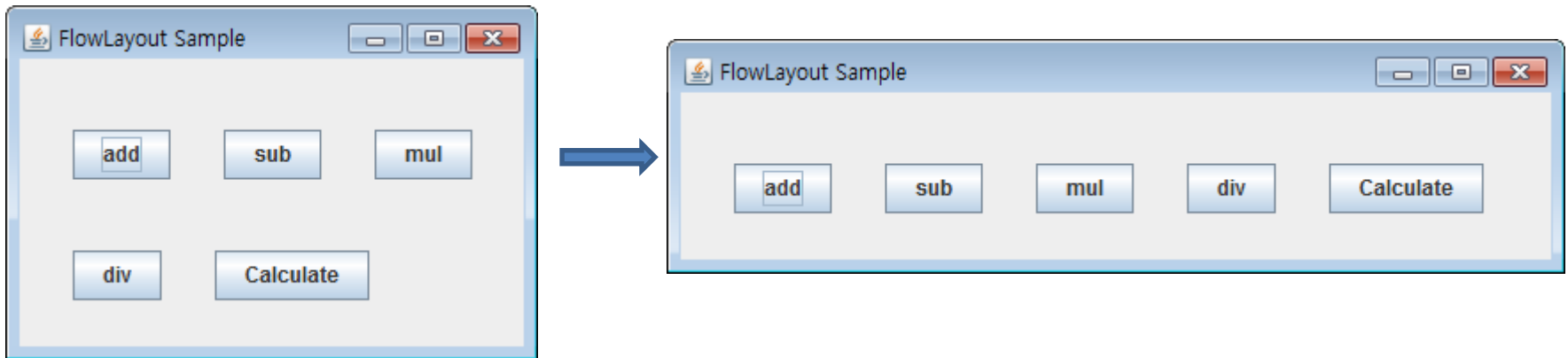
- 컨테이너 공간 내에 왼쪽에서 오른쪽으로 배치
 - 다시 위에서 아래로 순서대로 컴포넌트를 배치한다.



```
container.setLayout(new FlowLayout());  
container.add(new JButton("add"));  
container.add(new JButton("sub"));  
container.add(new JButton("mul"));  
container.add(new JButton("div"));  
container.add(new JButton("Calculate"));
```

FlowLayout

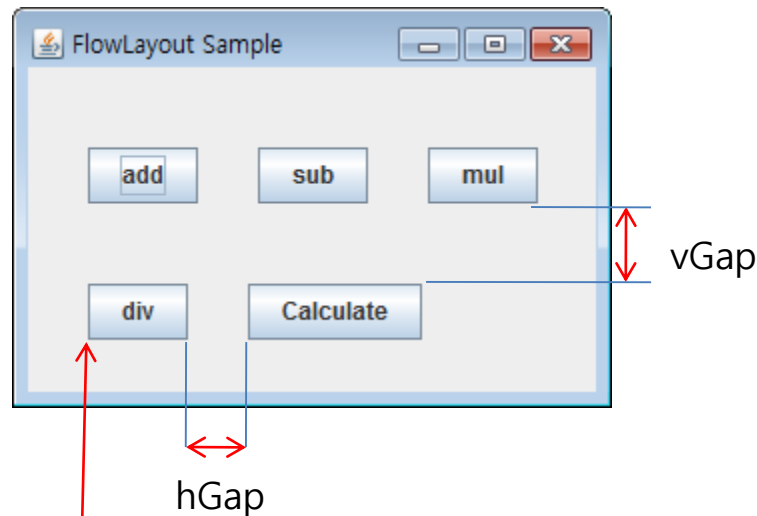
- 컨테이너의 크기가 변하면 배치 관리자에 의해 재배치됨



프레임의 크기를 바꾸면 배치도 변한다.

FlowLayout - 생성자와 속성

- 생성자
 - FlowLayout()
 - FlowLayout(int align)
 - FlowLayout(int align, int hGap, int vGap)
 - align : 컴포넌트의 정렬(5 가지중 많이 사용되는 3 가지)
 - FlowLayout.LEFT, FlowLayout.RIGHT, FlowLayout.CENTER(디폴트)
 - hGap : 좌우 두 컴포넌트 사이의 수평 간격, 픽셀 단위(디폴트 : 5)
 - vGap : 상하 두 컴포넌트 사이의 수직 간격, 픽셀 단위(디폴트 : 5)



FlowLayout.LEFT로 정렬됨

예제 : LEFT로 정렬되는 수평 간격이 30 픽셀, 수직 간격이 40 픽셀인 FlowLayout 사용 예

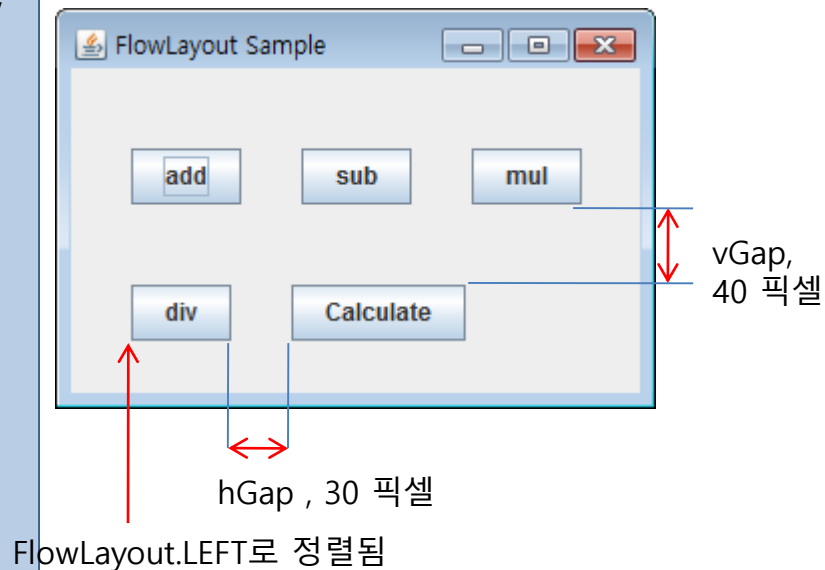
```
import javax.swing.*;
import java.awt.*;

public class FlowLayoutEx extends JFrame {
    FlowLayoutEx() {
        setTitle("FlowLayout Sample");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new FlowLayout(FlowLayout.LEFT, 30, 40));
        add(new JButton("add"));
        add(new JButton("sub"));
        add(new JButton("mul"));
        add(new JButton("div"));
        add(new JButton("Calculate"));

        setSize(300, 250);
        setVisible(true);
    }

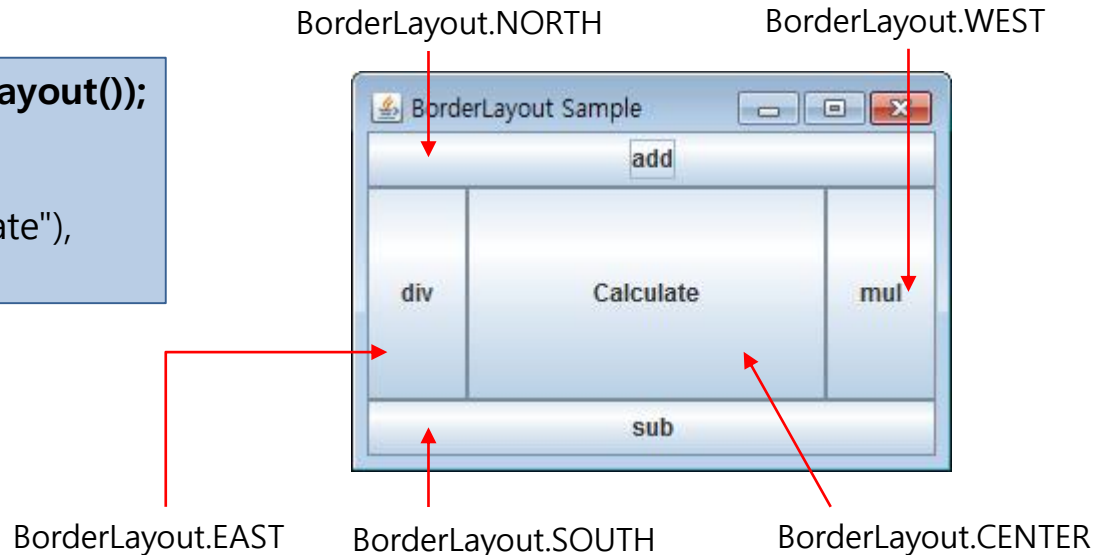
    public static void main(String[] args) {
        new FlowLayoutEx();
    }
}
```



BorderLayout

- 배치 방법
 - 컨테이너 공간을 5 구역으로 분할, 배치
 - East, West, South, North, Center
 - 배치 방법
 - `add(Component comp, int index)`
 - `comp`를 `index`의 공간에 배치
 - 컨테이너의 크기가 변하면 재배포

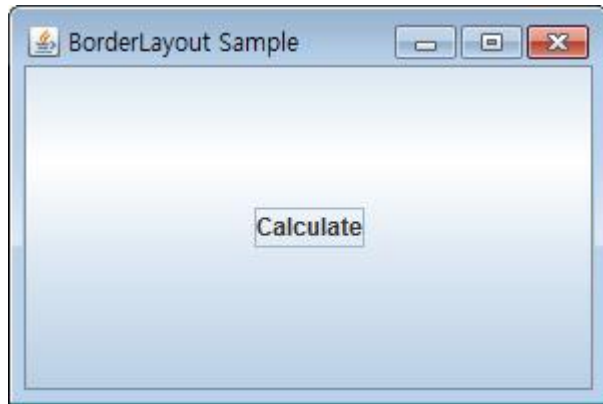
```
container.setLayout(new BorderLayout());  
container.add(new JButton("div"),  
              BorderLayout.WEST);  
container.add(new JButton("Calculate"),  
              BorderLayout.CENTER);
```



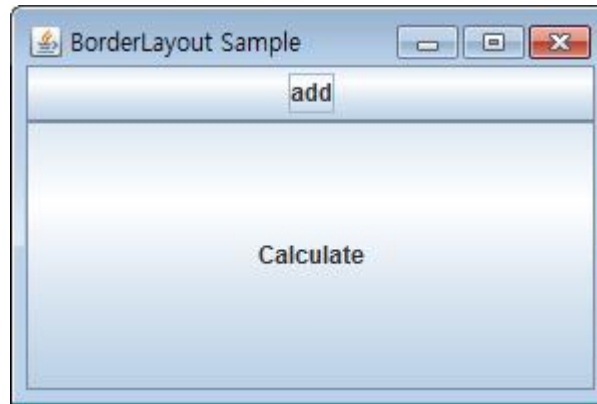
BorderLayout 생성자와 속성

- 생성자
 - BorderLayout()
 - BorderLayout(int hGap, int vGap)
 - hGap : 좌우 두 컴포넌트 사이의 수평 간격, 픽셀 단위(디폴트 : 0)
 - vGap : 상하 두 컴포넌트 사이의 수직 간격, 픽셀 단위(디폴트 : 0)

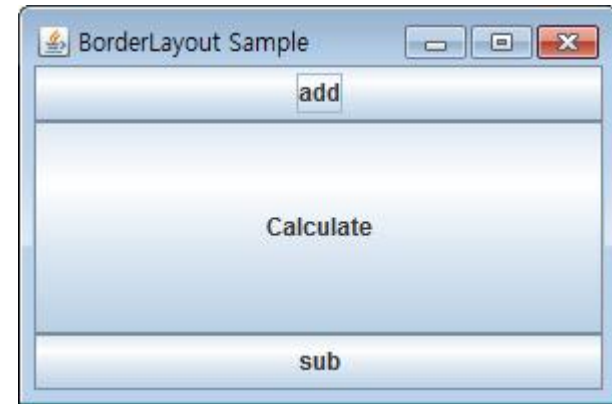
BorderLayout의 사용예



CENTER에 컴포넌트가 삽입될 때

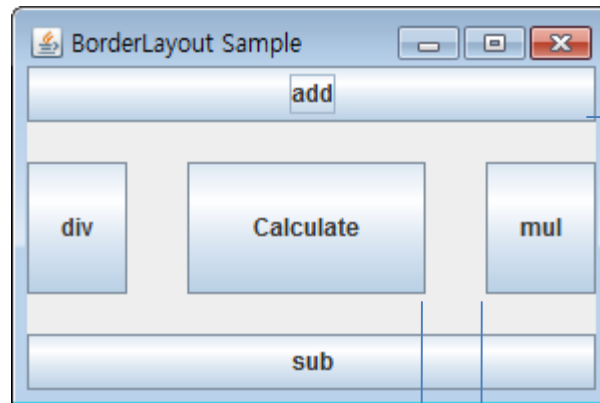


CENTER와 NORTH에 컴포넌트가 삽입될 때



CENTER, NORTH, SOUTH에
컴포넌트가 삽입될 때

new BorderLayout(30,20);
으로 배치관리자를
생성하였을 때



hGap , 30 픽셀

vGap,
20 픽셀

예제 : BorderLayout 사용 예

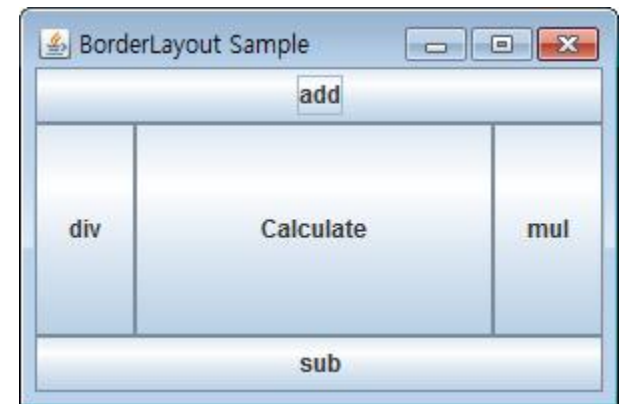
```
import javax.swing.*;
import java.awt.*;

public class BorderLayoutEx extends JFrame {
    BorderLayoutEx() {
        setTitle("BorderLayout Sample");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new BorderLayout());
        add(new JButton("add"), BorderLayout.NORTH);
        add(new JButton("sub"), BorderLayout.SOUTH);
        add(new JButton("mul"), BorderLayout.EAST);
        add(new JButton("div"), BorderLayout.WEST);
        add(new JButton("Calculate"), BorderLayout.CENTER);

        setSize(300, 200);
        setVisible(true);
    }

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



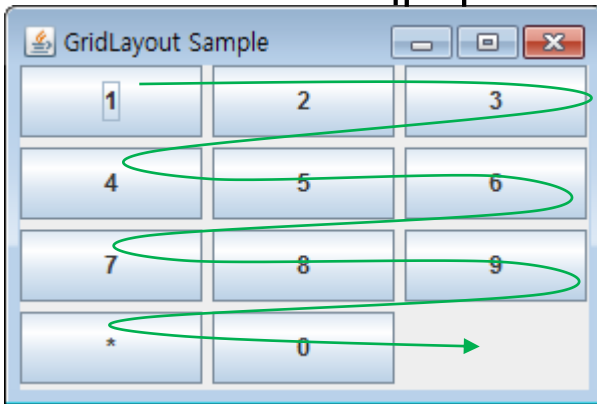
예제: LayoutBorder 만들기 예

- LayoutBorder1.java
 - BorderLayout 으로 contentpane을 만듦

GridLayout

- 배치방법

- 컨테이너 공간을 동일한 사각형 격자(그리드)로 분할하고 각 셀에 하나의 컴포넌트 배치
 - 격자 구성은 생성자에 행수와 열수 지정
 - 셀에 왼쪽에서 오른쪽으로, 다시 위에서 아래로 순서대로 배치



```
container.setLayout(new GridLayout(4,3,5,5)); // 4×3 분할로 컴포넌트 배치  
container.add(new JButton("1")); // 상단 왼쪽 첫 번째 셀에 버튼 배치  
container.add(new JButton("2")); // 그 옆 셀에 버튼 배치
```

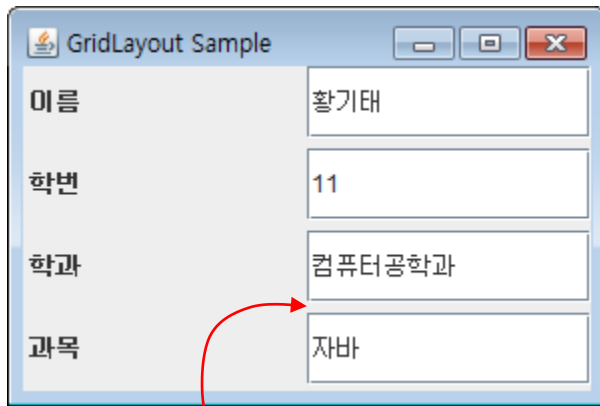
- 4x3 그리드 레이아웃 설정
- 총 11 개의 버튼이 순서대로 add 됨
- 수직 간격 vGap : 5 픽셀
- 수평 간격 hGap : 5 픽셀

- 컨테이너의 크기가 변하면 재배포
 - 크기 재조정

GridLayout 생성자와 속성

- 생성자
 - GridLayout()
 - GridLayout(int rows, int cols)
 - GridLayout(int rows, int cols, int hGap, int vGap)
 - rows : 격자의 행수 (디폴트 : 1)
 - cols : 격자의 열수 (디폴트 : 1)
 - hGap : 좌우 두 컴포넌트 사이의 수평 간격, 픽셀 단위 (디폴트 : 0)
 - vGap : 상하 두 컴포넌트 사이의 수직 간격, 픽셀 단위 (디폴트 : 0)
 - rows x cols 만큼의 셀을 가진 격자로 컨테이너 공간을 분할, 배치

예제 : GridLayout으로 입력 폼 만들기



두 행 사이의 수직 간격
vGap이 5 픽셀로 설정됨

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

public class GridLayoutEx extends JFrame {
    GridLayoutEx() {
        setTitle("GridLayout Sample");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        GridLayout grid = new GridLayout(4, 2);
        grid.setVgap(5);
        setLayout(grid);
        add(new JLabel(" 이름"));
        add(new JTextField(""));
        add(new JLabel(" 학번"));
        add(new JTextField(""));
        add(new JLabel(" 학과"));
        add(new JTextField(""));
        add(new JLabel(" 과목"));
        add(new JTextField(""));

        setSize(300, 200);
        setVisible(true);
    }

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

예제: GridLayout 만들기 예

- GridLayout2.java
 - GridLayout 으로 contentpane을 만듦
 - 레벨을 배치함
- GridLayout2.java
 - 버튼을 배치함
- LayoutComplex1.java
 - 여러가지 조합하여 만듦
- LayoutCard1.java
 - 화면이 전환되는 레이아웃 만듦

배치관리자 없는 컨테이너

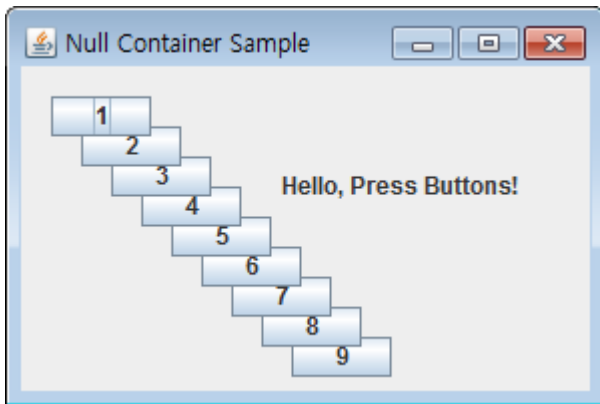
- 배치관리자가 없는 컨테이너 개념
 - 응용프로그램에서 컴포넌트의 절대 크기와 절대 위치 결정
- 용도
 - 컴포넌트의 크기나 위치를 개발자 임의로 결정하고자 하는 경우
 - 게임 프로그램과 같이 시간이나 마우스/키보드의 입력에 따라 컴포넌트들의 위치와 크기가 수시로 변하는 경우
 - 여러 컴포넌트들이 서로 겹쳐 출력하고자 하는 경우
- 컨테이너의 배치 관리자 제거 방법
 - `container.setLayout(null);`

// JPanel의 배치관리자를 삭제하는 예

```
JPanel p = new JPanel();  
p.setLayout(null);
```

- 컴포넌트의 크기와 위치 설정
 - 프로그램 내에서 이루어져야 함
 - 컴포넌트들이 서로 겹치게 할 수 있음
 - 다음 메소드 이용
 - 컴포넌트 크기 설정 : `component.setSize(int width, int height);`
 - 컴포넌트 위치 설정 : `component.setLocation(int x, int y);`
 - 컴포넌트 위치와 크기 동시 설정 : `component.setBounds(int x, int y, int width, int height);`

예제 : 배치관리자 없는 컨테이너에 컴포넌트 위치와 크기를 절대적으로 지정



원하는 위치에 원하는 크기로
컴포넌트를 마음대로
배치할 수 있다.

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

public class NullContainerEx extends JFrame {
    NullContainerEx() {
        setTitle("Null Container Sample");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);

        JLabel la = new JLabel("Hello, Press Buttons!");
        la.setLocation(130, 50);
        la.setSize(200, 20);
        add(la);
        for(int i=1; i<=9; i++) {
            JButton b = new JButton(Integer.toString(i));
            b.setLocation(i*15, i*15);
            b.setSize(50, 20);
            add(b);
        }
        setSize(300, 200);
        setVisible(true);
    }

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

여러가지 컴포넌트



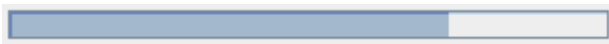
JToolBar



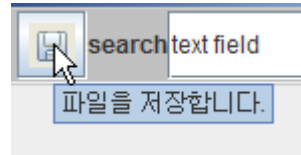
JTabbedPane



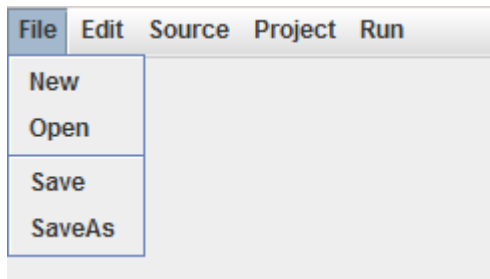
JSplitPane



JProgressBar



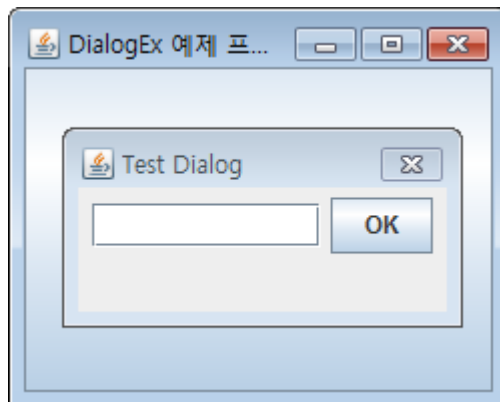
JToolTip



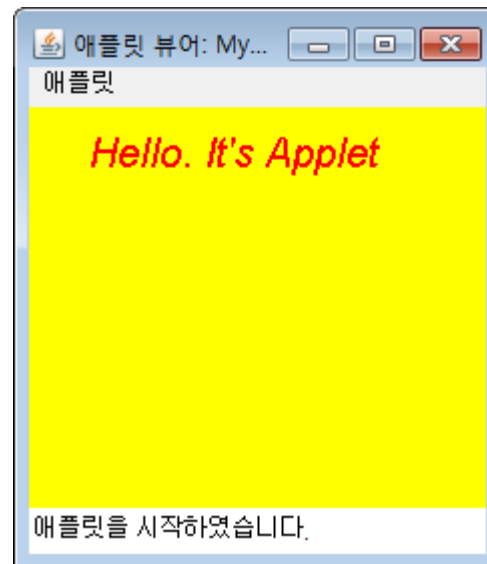
JMenu



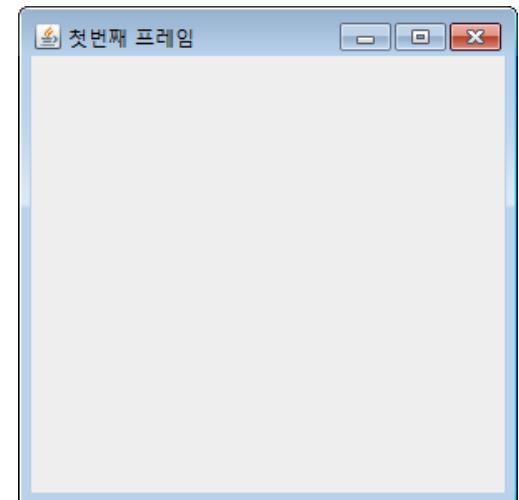
JScrollPane



JDialog



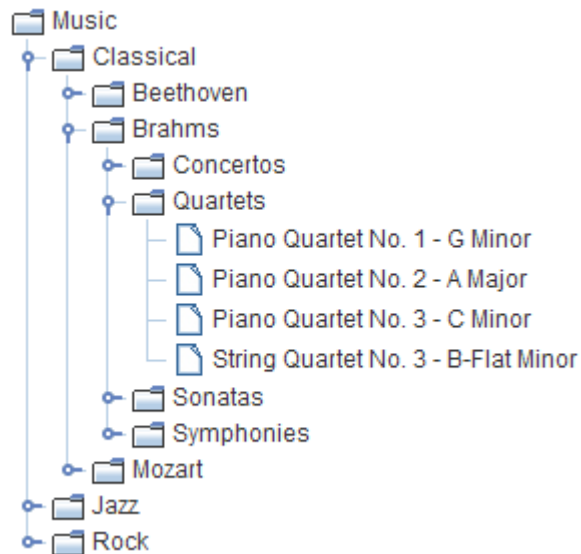
JApplet



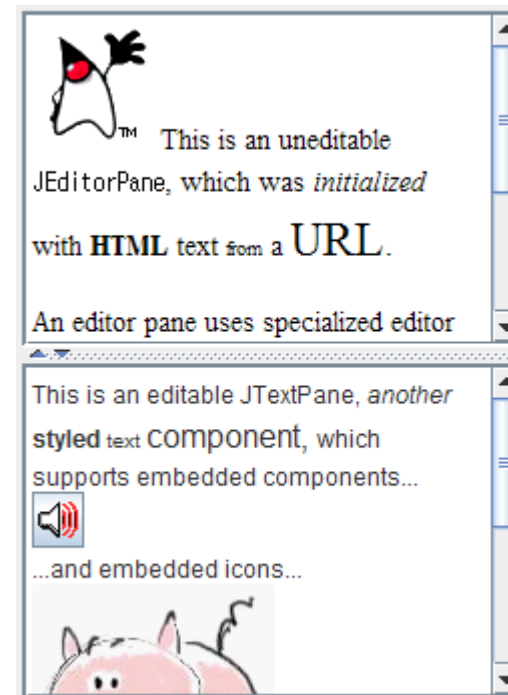
JFrame

| First Name | Last Name | Favorite Color | Favorite Movie | Favorite Number | Favorite Food |
|------------|-----------|----------------|-----------------------|-----------------|---|
| Mike | Albers | Green | Brazil | 44 |  |
| Mark | Andrews | Blue | Curse of the Dem... | 3 |  |
| Brian | Beck | Black | The Blues Brothers | 2.718 |  |
| Lara | Bunni | Red | Airplane (the whol... | 15 |  |
| Roger | Brinkley | Blue | The Man Who Kn... | 13 |  |
| Brent | Christian | Black | Blade Runner (Dir... | 23 |  |

JTable



JTree



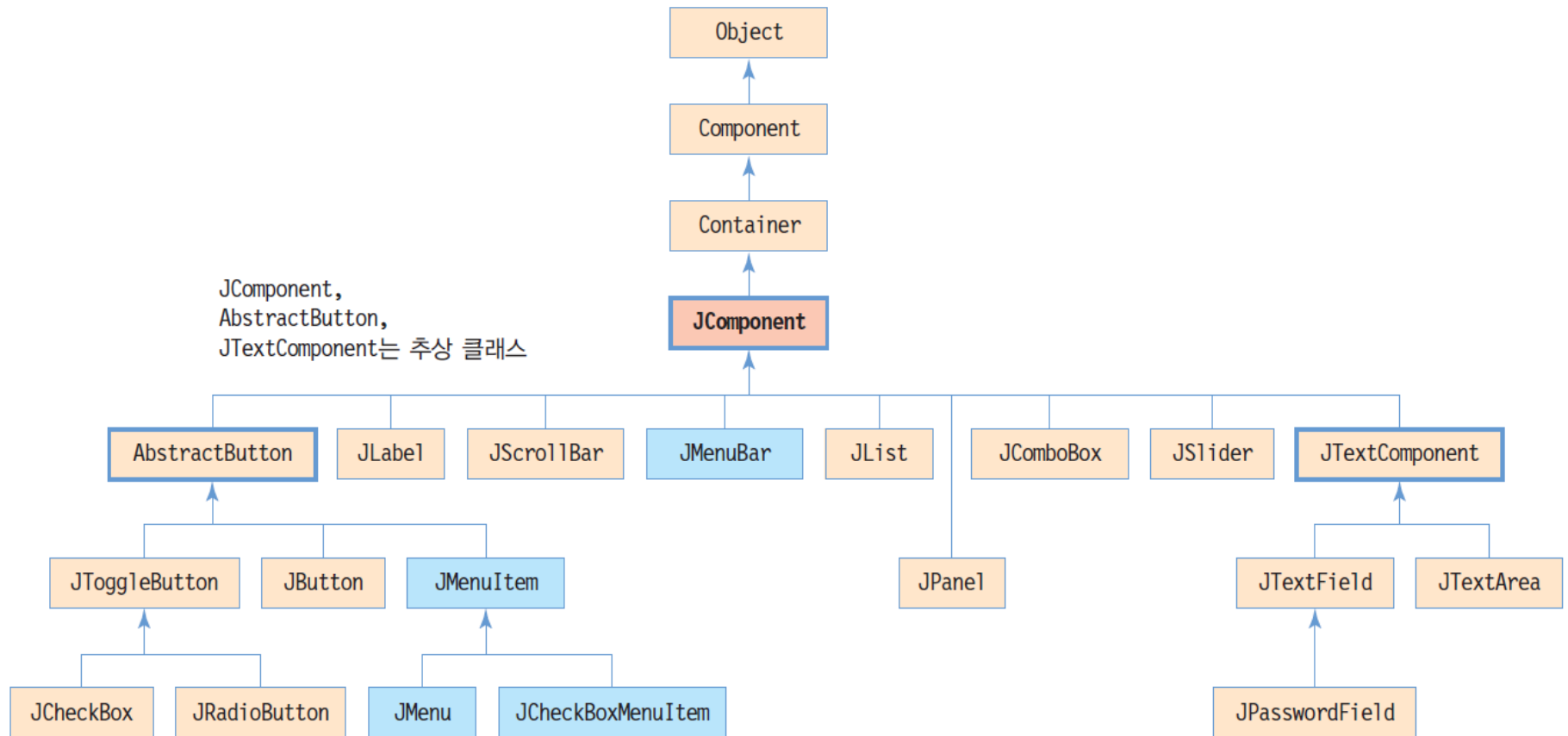
JEditorPane and JTextPane

Swing 으로 만든 GUI 프로그램 샘플



3. Choice, Menu

기초적인 스윙 컴포넌트와 상속 관계



스윙 컴포넌트의 공통 메소드, JComponent의 메소드

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 리스트 리턴  
Container getParent() 부모 컨테이너 리턴  
Container getTopLevelAncestor() 최상위 부모 컨테이너 리턴
```

스윙 컴포넌트의 공통 메소드 확인 사례

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

```
public class SwingAPIEx extends JFrame {
```

```
    Container contentPane;
    JLabel la;
    JButton b1, b2, b3, b4;
```

```
    SwingAPIEx() {
```

```
        setTitle("Swing 공통 메소드 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());
```

```
        b1 = new JButton("위치와 크기 정보");
        b1.addActionListener(new MyButtonListener());
        contentPane.add(b1);
```

```
        b2 = new JButton("모양 정보");
        b2.setOpaque(true);
        b2.setForeground(Color.MAGENTA);
        b2.setBackground(Color.YELLOW);
        b2.setFont(new Font("고딕체", Font.ITALIC, 20));
        b2.addActionListener(new MyButtonListener());
        contentPane.add(b2);
```

```
        b3 = new JButton("작동하지 않는 버튼");
        b3.setEnabled(false);
        b3.addActionListener(new MyButtonListener());
        contentPane.add(b3);
```

```
        b4 = new JButton("숨기기/보이기");
        b4.addActionListener(new MyButtonListener());
        contentPane.add(b4);
```

```
        setSize(250,200);
        setVisible(true);
```

```
    }
```

```
class MyButtonListener implements ActionListener {
```

```
    public void actionPerformed(ActionEvent e) {
```

```
        Object source = e.getSource();
```

```
        if(source == b1) {
```

```
            System.out.println("버튼의 위치와 크기");
```

```
            System.out.println("위치 = (" + b1.getX() + "," + b1.getY() + ")");
```

```
            System.out.println("크기 = (" + b1.getWidth() + "x"
                                + b1.getHeight() + ")");
```

```
            JPanel c = (JPanel)b2.getParent();
```

```
            System.out.println("컨텐츠판의 위치와 크기");
```

```
            System.out.println("위치 = (" + c.getX() + "," + c.getY() + ")");
```

```
            System.out.println("크기 = (" + c.getWidth() + "x"
                                + c.getHeight() + ")");
```

```
        }
```

```
        else if(source == b2) {
```

```
            System.out.println("폰트 = " + b2.getFont());
```

```
            System.out.println("배경색 = " + b2.getBackground());
```

```
            System.out.println("글자색 = " + b2.getForeground());
```

```
        }
```

```
        else {
```

```
            if(b1.isVisible()) {
```

```
                b1.setVisible(false);
```

```
                b2.setVisible(false);
```

```
                b3.setVisible(false);
```

```
            }
```

```
            else {
```

```
                b1.setVisible(true);
```

```
                b2.setVisible(true);
```

```
                b3.setVisible(true);
```

```
            }
```

```
        }
```

```
    }
```

```
}
```

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

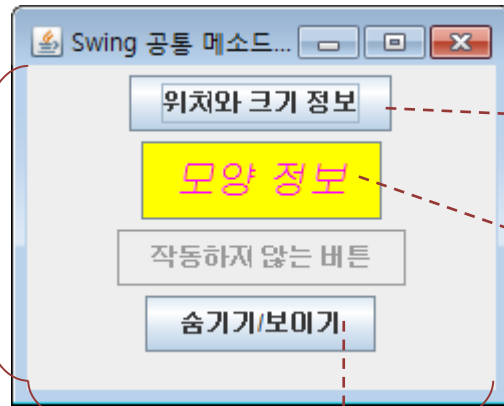
```
    new SwingAPIEx();
```

```
}
```

```
}
```

실행: 스윙 컴포넌트의 공통 요소

콘솔 창에 출력된 내용



컨텐츠팬
의 높이
164 픽셀

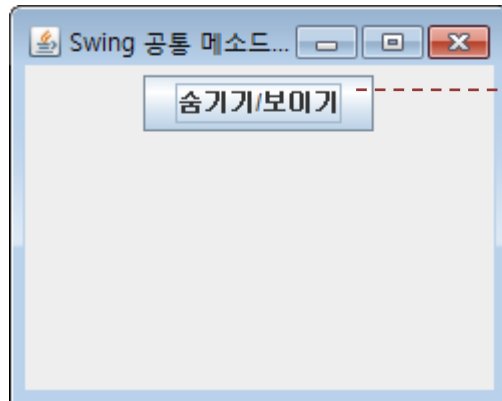
버튼을 선택한 경우

버튼을 선택한 경우

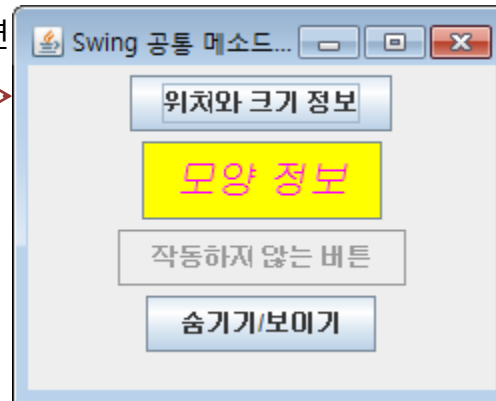
```
버튼의 위치와 크기  
위치 = (51,5)  
크기 = (131x28)  
컨텐츠팬의 위치와 크기  
위치 = (0,0)  
크기 = (234x164)  
폰트 = java.awt.Font[family=Dialog,name=고딕체,style=italic,size=20]  
배경색 = java.awt.Color[r=255,g=255,b=0]  
글자색 = java.awt.Color[r=255,g=0,b=255]
```

컨텐츠팬의 폭, 234 픽셀

버튼을 선택하면 나머지
3개의 버튼이 보이지
않게 됨



버튼을 선택하면
다시 보이게 됨



JLabel, 레이블 컴포넌트

- JLabel의 용도
 - 텍스트나 이미지를 컴포넌트화 하여 출력하기 위한 목적
- 레이블 컴포넌트 생성
 - JLabel()
 - 빈 레이블 컴포넌트 생성
 - JLabel(Icon image)
 - 이미지만을 가진 레이블 컴포넌트 생성
 - JLabel(String text)
 - 텍스트만을 가진 레이블 컴포넌트 생성
 - JLabel(String text, Icon image, int hAlignment)
 - 텍스트와 이미지, 수평 정렬 값을 가진 레이블 컴포넌트 생성
 - 수평 정렬 값 hAlignment에 사용 가능한 값들 :
 - SwingConstants.LEFT, CENTER, RIGHT, LEADING or TRAILING

레이블 컴포넌트 생성 예

- 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 : JLabel 컴포넌트 생성 예

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

public class LabelEx extends JFrame {
    Container contentPane;
    LabelEx() {
        setTitle("레이블 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.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/normalcon.gif");
        JLabel label = new JLabel("보고싶으면 전화하세요",
            normalIcon, SwingConstants.CENTER);

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

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



JButton, 버튼 컴포넌트

- 버튼 컴포넌트
 - 버튼 모양의 컴포넌트
 - 버튼은 클릭될 때 Action 이벤트를 발생시킴
- 버튼 컴포넌트 생성
 - JButton()
 - 빈 버튼 생성
 - JButton(Icon icon)
 - 이미지 아이콘만 가진 버튼 생성
 - JButton(String text)
 - 텍스트만 가진 버튼 생성
 - JButton(String text, Icon icon)
 - 텍스트와 이미지 아이콘 모두 가진 버튼 생성
- 버튼 컴포넌트 생성 예
 - "hello" 문자열을 가진 버튼 컴포넌트 생성 예



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

이미지 버튼 컴포넌트 만들기

- 하나의 버튼에 3 개의 이미지 연결
 - 사용자의 마우스 접근에 따라 3 개의 이미지 중 선택 출력
- 3 개의 버튼 이미지
 - 버튼의 보통 상태 때 출력되는 이미지 : `normalIcon`
 - 생성자에 이미지 아이콘 전달
 - 버튼에 마우스가 올라갈 때 출력되는 이미지 : `rolloverIcon`
 - 이미지 설정 메소드 : `JButton.setRolloverIcon(icon);`
 - 버튼을 누른 상태 때 출력되는 이미지 : `pressedIcon`
 - 이미지 설정 메소드 : `JButton.setPressedIcon(icon)`
- 이미지 아이콘 생성
 - `new ImageIcon(이미지 경로명);`
 - `new ImageIcon("images/normalIcon.gif");`

예제 : 3 개의 이미지 아이콘을 가진 버튼 만들기

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

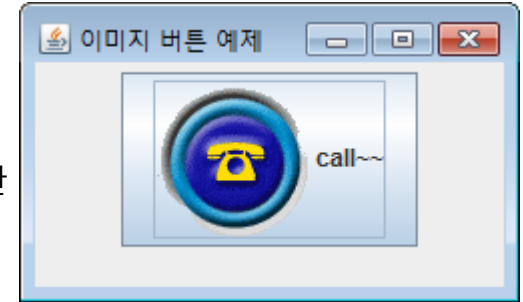
public class ButtonImageEx extends JFrame {
    Container contentPane;
    ButtonImageEx() {
        setTitle("버튼에 아이콘 달기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());

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

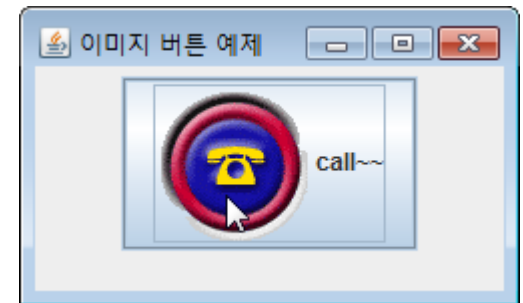
        JButton btn = new JButton("call~~", normalIcon);
        btn.setRolloverIcon(rolloverIcon);
        btn.setPressedIcon(pressedIcon);
        contentPane.add(btn);

        setSize(250,200);
        setVisible(true);
    }
    public static void main(String [] args) {
        new ButtonImageEx();
    }
}
```

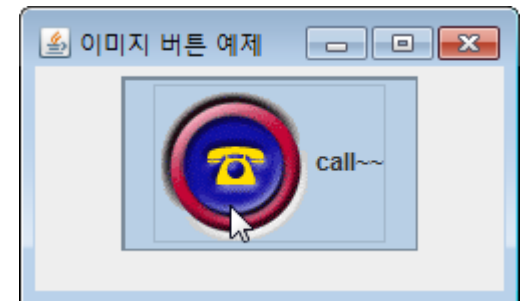
보통 상태에 있는 동안
(normalIcon.gif)



마우스가 버튼 위에
올라간 경우
(rolloverIcon.gif)



마우스가 눌려진 순간
(pressedIcon.gif)



레이블과 버튼의 정렬(Alignment)

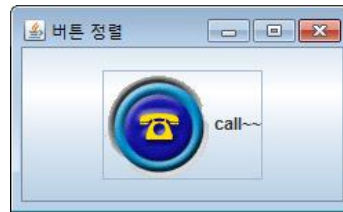
- 수평 정렬
 - 컴포넌트 영역 내에 이미지와 텍스트의 수평 위치
 - `void setHorizontalAlignment(int align)`

버튼 영역



왼쪽정렬

`SwingConstants.LEFT`



중앙정렬

`SwingConstants.CENTER`

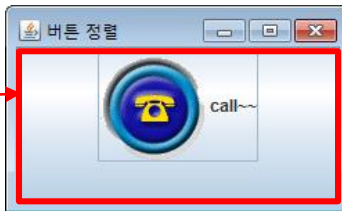


오른쪽정렬

`SwingConstants.RIGHT`

- 수직 정렬
 - 컴포넌트 영역 내에 콘텐츠(이미지와 텍스트)의 수직 위치
 - `void setVerticalAlignment(int align)`

버튼 영역



위쪽정렬

`SwingConstants.TOP`



중앙정렬

`SwingConstants.CENTER`

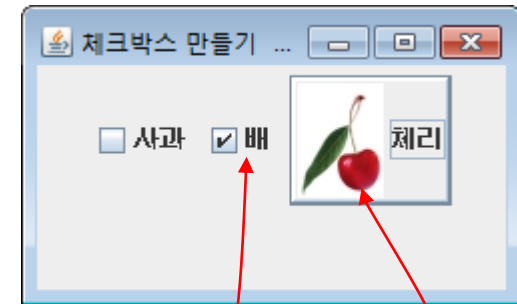


아래쪽정렬

`SwingConstants.BOTTOM`

JCheckBox, 체크박스 컴포넌트

- 체크박스
 - 선택(selected)과 비선택(deselected)의 두 상태만 가지는 버튼
- 생성자
 - 디폴트는 선택되지 않은 상태
 - JCheckBox ()
 - 텍스트와 이미지가 없는 토글 버튼 생성
 - JCheckBox(Icon icon)
 - 이미지만 가진 토글 버튼 생성
 - JCheckBox(Icon icon, boolean selected)
 - 이미지와 지정된 선택 상태로 생성
 - JCheckBox(String text)
 - 텍스트 만 가진 토글 버튼 생성
 - JCheckBox(String text, boolean selected)
 - 텍스트와 지정된 선택 상태로 생성
 - JCheckBox(String text, Icon icon)
 - 텍스트와 이미지 둘 다 가진 토글 버튼 생성
 - JCheckBox(String text, Icon icon, boolean selected)
 - 텍스트와 이미지를 가지고 지정된 선택상태로 생성



체크박스 문자열

체크박스 이미지

체크 박스 생성


- 텍스트 정보만을 가진 체크 박스 생성

- "사과" 텍스트를 가진 체크박스 생성


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

- "배" 텍스트를 가지고 선택상태로 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

- 선택 상태의 이미지를 위해 selectedCherry.jpg를 사용하였음

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

예제 : 체크박스 생성 예

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

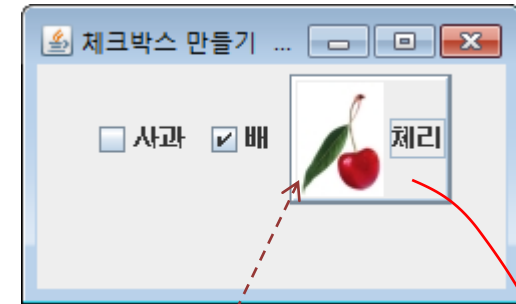
public class CheckBoxEx extends JFrame {
    Container contentPane;
    CheckBoxEx() {
        setTitle("체크박스 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());

        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);

        contentPane.add(apple);
        contentPane.add(pear);
        contentPane.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)

JCheckBox와 Item 이벤트

- Item 이벤트

- 체크 박스가 선택되거나 해제되는 경우에 발생하는 이벤트
 - 사용자가 마우스나 키보드로 체크박스를 선택하거나 해제한 경우
 - 프로그램에서 체크박스를 선택하거나 해제한 경우

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

- 체크박스 컴포넌트의 모양을 변경한 후 이벤트 리스너 호출
- ItemEvent 객체 생성

- ItemListener 인터페이스의 추상 메소드

- public void itemStateChanged(ItemEvent e)

- ItemEvent의 주요 메소드

- int getStateChange()
 - 체크 박스의 상태가 선택 상태인지 해제상태인지 리턴
 - ItemEvent.SELECTED 또는 ItemEvent.DESELECTED
- Object getItem()
 - 이벤트를 발생시킨 아이템 객체
 - 체크박스의 경우 이벤트가 발생한 JCheckBox 객체 리턴

예제 : ItemEvent를 활용하여 가격 합산하기

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

public class CheckBoxItemEventEx extends JFrame {
    Container contentPane;
    JCheckBox [] fruits = new JCheckBox [3];
    String [] names = {"사과", "배", "체리"};
    JLabel sumLabel;
    int sum = 0;

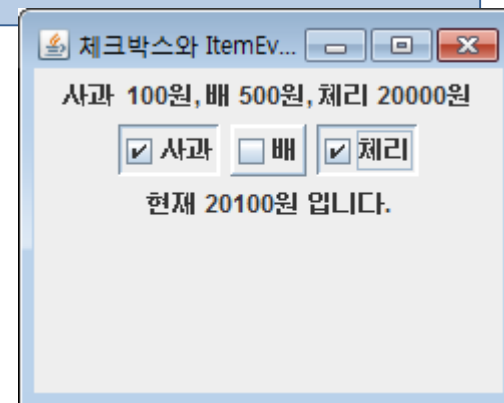
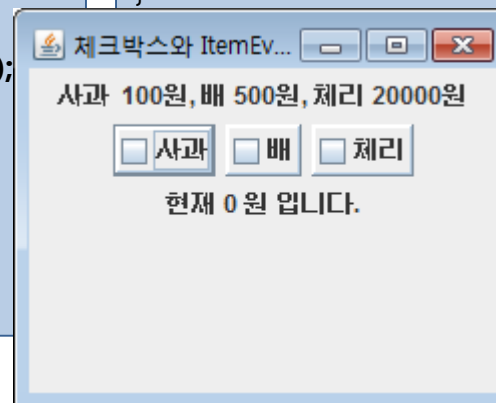
    CheckBoxItemEventEx() {
        setTitle("체크박스와 ItemEvent 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());

        contentPane.add(
            new JLabel("사과 100원, 배 500원, 체리 20000원"));
        for(int i=0; i<fruits.length; i++) {
            fruits[i] = new JCheckBox(names[i]);
            fruits[i].setBorderPainted(true);
            contentPane.add(fruits[i]);
            fruits[i].addItemListener(new MyItemListener());
        }
        sumLabel = new JLabel("현재 0 원 입니다.");
        contentPane.add(sumLabel);
        setSize(250,200);
        setVisible(true);
    }
}
```

```
class MyItemListener implements ItemListener {
    public void itemStateChanged(ItemEvent e) {
        int selected=1;
        if(e.getStateChange() == ItemEvent.SELECTED)
            selected = 1;
        else
            selected = -1;
        if(e.getItem() == fruits[0])
            sum = sum + selected*100;
        else if(e.getItem() == fruits[1])
            sum = sum + selected*500;
        else
            sum = sum + selected*20000;

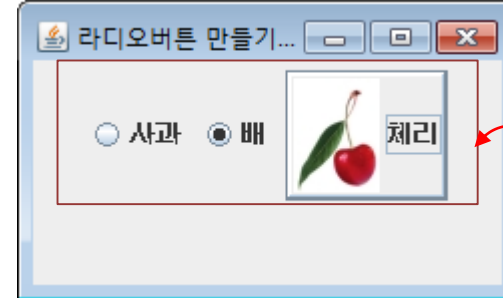
        sumLabel.setText("현재 "+sum+"원 입니다.");
    }
}

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



라디오버튼, JRadioButton

- 라디오버튼이란?
 - 여러 버튼으로 그룹을 형성하고, 그룹에 속한 버튼 중 하나만 선택되는 버튼
 - 다른 버튼이 선택되면 이전에 선택된 버튼은 자동으로 해제됨
 - 체크박스와의 차이점
 - 체크 박스는 각 체크박스마다 선택/해제가 가능하지만 라디오 버튼은 그룹에 속한 버튼 중 하나만 선택 상태가 됨
 - 이미지를 가진 라디오버튼의 생성 및 다루기는 체크박스와 완전히 동일
- 생성자
 - 디폴트는 선택되지 않은 상태, JCheckBox의 생성자와 동일
 - JRadioButton()
 - 텍스트와 이미지가 없는 토글 버튼 생성
 - JRadioButton(Icon icon)
 - 이미지만 가진 토글 버튼 생성
 - JRadioButton(Icon icon, boolean selected)
 - 이미지와 지정된 선택 상태로 생성
 - JRadioButton(String text)
 - 텍스트 만 가진 토글 버튼 생성
 - JRadioButton(String text, boolean selected)
 - 텍스트와 지정된 선택 상태로 생성
 - JRadioButton(String text, Icon icon)
 - 텍스트와 이미지 둘 다 가진 토글 버튼 생성
 - JRadioButton(String text, Icon icon, boolean selected)
 - 텍스트와 이미지를 가지고 지정된 선택상태로 생성

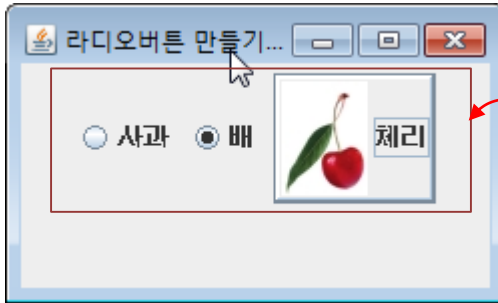


하나의 버튼 그룹에 속한 라디오버튼들

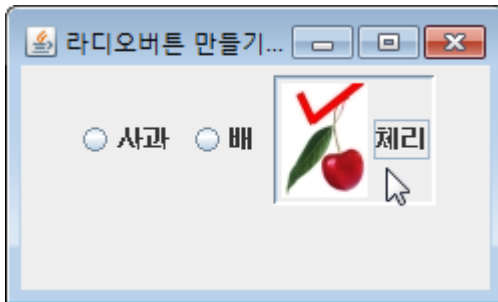
라디오 버튼 생성 과정

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);`

라디오버튼 생성 예



초기 상태(배가 선택된 상태)



체리가 선택된 상태

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

```
public class RadioButtonEx extends JFrame {
    Container contentPane;
    RadioButtonEx() {
        setTitle("라디오버튼 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.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);

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

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

예제: ItemEvent 활용, 사진 보여 주기

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

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

    RadioButtonItemEventEx() {
        setTitle("라디오버튼 Item Event 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new BorderLayout());

        JPanel panel = new JPanel();
        panel.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]);
            panel.add(radio[i]);
            radio[i].addItemListener(new MyItemListener());
        }
        radio[2].setSelected(true);
        contentPane.add(panel, BorderLayout.NORTH);
        contentPane.add(imageLabel, BorderLayout.CENTER);
        imageLabel.setHorizontalAlignment(SwingConstants.CENTER);

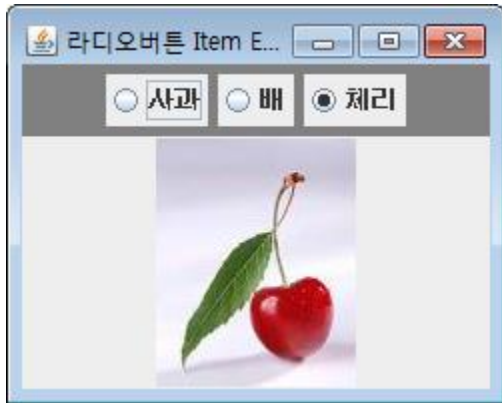
        setSize(250,200);
        setVisible(true);
    }
}
```

```
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]);
    }
}

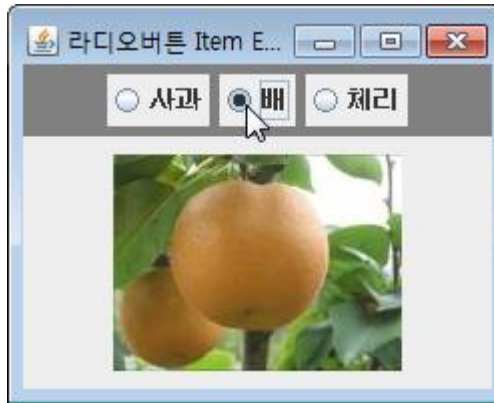
public static void main(String [] args) {
    new RadioButtonItemEventEx();
}
```

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

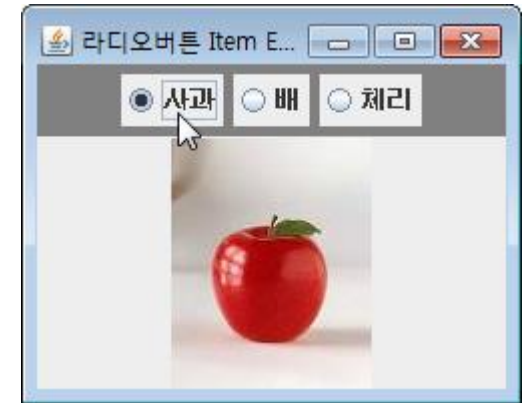
예제 실행: ItemEvent 활용, 사진 보여 주기



초기화면



"배"를 선택한 경우



"사과"를 선택한 경우

예제: 라디오, 체크박스 사용하기 예

- SwingRadio1.java
- SwingCheck1.java

JTextField, 텍스트필드 컴포넌트

- 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 입력 창 생성
 - JTextField(int cols)
 - 입력 창의 크기가 cols 개, 빈 입력 창 생성
 - JTextField(String text)
 - text 문자열로 초기화된 입력 창 생성
 - JTextField(String text, int cols)
 - 입력 창의 크기가 cols 개이고, text 문자열이 초기 출력된 텍스트 입력 창 생성

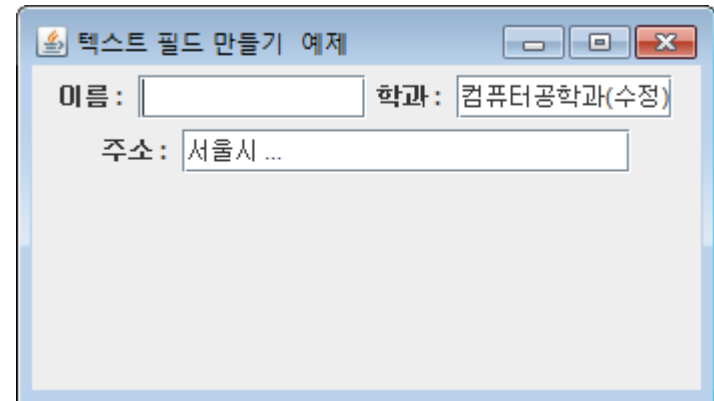
예제 : 간단한 텍스트 필드 만들기

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

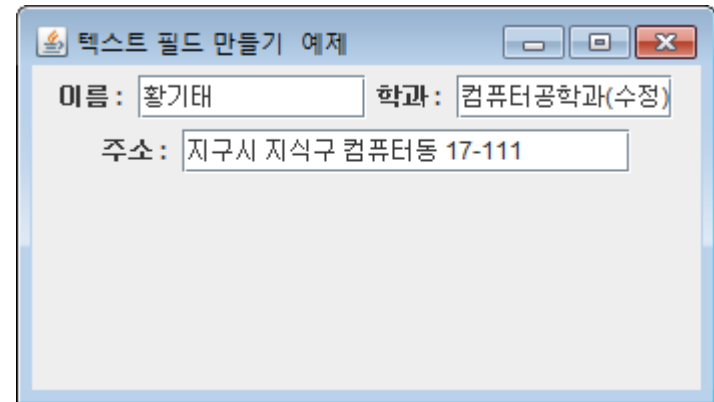
public class SwingTextField1 extends JFrame {
    Container contentPane;
    SwingTextField1() {
        setTitle("텍스트 필드 만들기 예제");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());

        contentPane.add(new JLabel("이름 : "));
        contentPane.add(new JTextField(10));
        contentPane.add(new JLabel("학과 : "));
        contentPane.add(new JTextField("xxx 공학과"));
        contentPane.add(new JLabel("주소 : "));
        contentPane.add(new JTextField("서울시 ...", 20));
        setSize(300,200);
        setVisible(true);
    }

    public static void main(String [] args) {
        new SwingTextField1();
    }
}
```



초기화면



사용자가 입력한 경우

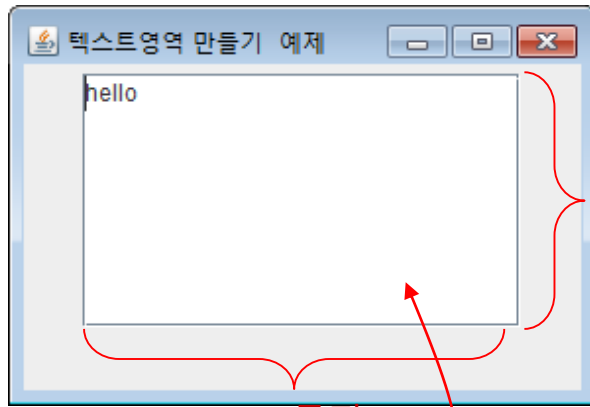
텍스트 필드의 주요 메소드

- 텍스트의 편집을 불가능하게 하기
 - `JTextField.setEditable(false);`
- 텍스트 창에 강제로 문자열 출력하기
 - `JTextField.setText("hello");`
- 텍스트 폰트 지정하기
 - `JTextField.setFont(new Font("고딕체", Font.ITALIC, 20);`
- 텍스트 창에 있는 문자열 선택하기
 - `JTextField.select(0, 5);` // 0번 문자에서 5번째까지 문자열 선택

TextArea, 텍스트 영역 컴포넌트

- 텍스트영역이란?
 - 여러 줄을 입력할 수 있는 텍스트 입력 창
 - 스크롤바를 지원하지 않는다.
 - JScrollPane 객체에 삽입하는 방식으로 스크롤바 지원
- 생성자
 - JTextArea()
 - 빈 입력 창 생성
 - JTextArea(int rows, int cols)
 - 크기가 rows x cols, 빈 입력 창 생성
 - JTextArea(String text)
 - text 문자열이 출력된 입력 창 생성
 - JTextArea(String text, int rows, int cols)
 - 크기가 rows x cols, text 문자열이 출력된 입력 창 생성

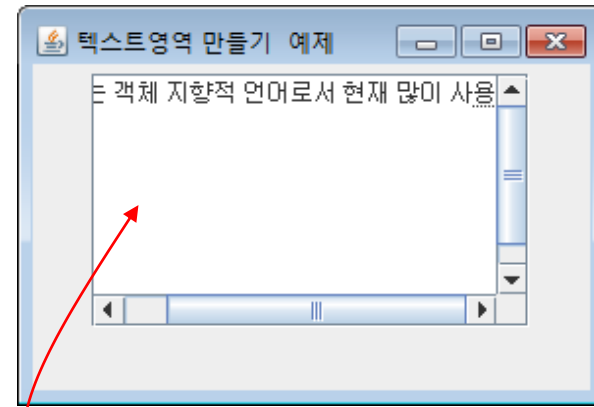
스크롤 가능한 텍스트영역 만들기



20 문자

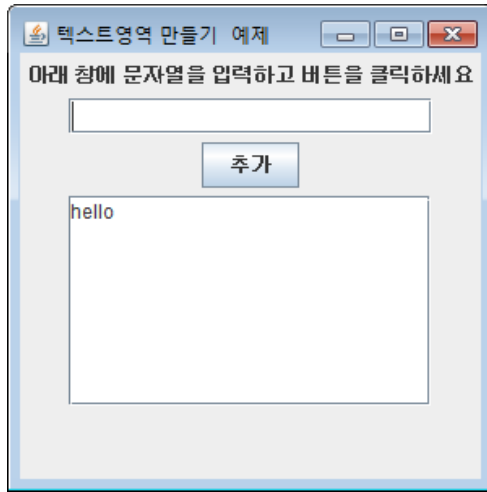
7 줄

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



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

JTextArea 생성 예



초기화면

텍스트필드에 입력 후
추가 버튼을 누른 경우



버튼이 선택되면 ta의
끝에 tf에 입력된 문자
열을 추가함

20x7 크기에 "hello"문
자열을 가진 JTextArea
컴포넌트 생성

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

```
public class TextAreaEx extends JFrame {
```

```
    Container contentPane;
```

```
    TextAreaEx() {
```

```
        setTitle("텍스트 영역 만들기 예제");
```

```
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
        contentPane = getContentPane();
```

```
        contentPane.add(new MyCenterPanel(),
                        BorderLayout.CENTER);
```

```
        setSize(300,300);
```

```
        setVisible(true);
```

```
    }
```

```
    class MyCenterPanel extends JPanel {
```

```
        JTextField tf;
```

```
        JButton btn;
```

```
        JTextArea ta;
```

```
        MyCenterPanel() {
```

```
            tf = new JTextField(20);
```

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

```
            btn.addActionListener(new ActionListener() {
```

```
                public void actionPerformed(ActionEvent e) {
```

```
                    ta.append(tf.getText()+"\n");
```

```
                }
```

```
            });
```

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

```
            add(new JLabel("아래 창에 문자열을 입력하고 버튼을 클릭하세요"));
```

```
            add(tf);
```

```
            add(btn);
```

```
            add(new JScrollPane(ta));
```

```
        }
```

```
    }
```

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

```
        new TextAreaEx();
```

```
    }
```

스크롤바를 출력하기
위해 JTextArea 컴포넌
트를 JScrollPane에 삽
입하고 JScrollPane 객
체를 패널에 삽입