Power Java

제17장 그래픽 사용자 인터페이스

이번 장에서 학습할 내용

- 그래픽 사용자 인터페이스
- 스윙의 소개
- 컨테이너
- GUI 작성 절차
- 기초 컴포넌트

그래픽 사용자 인터페이스

- 그래픽 사용자 인터페이스(Graphical User Interface, 간단히 GUI)
- 컴포넌트들로 구성

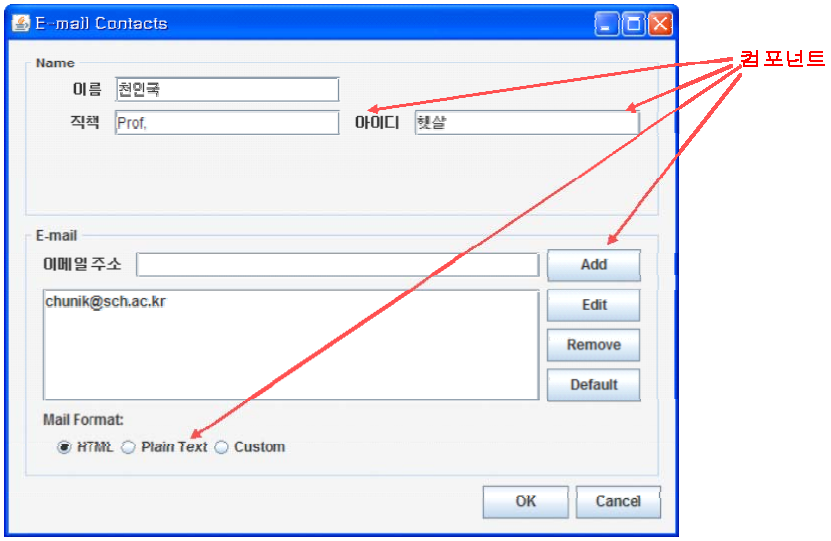
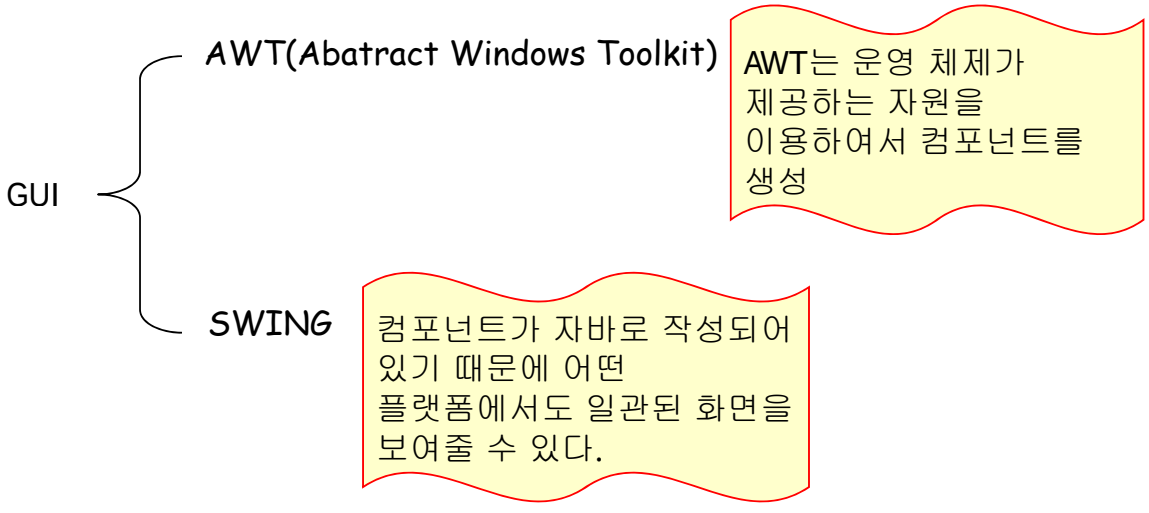
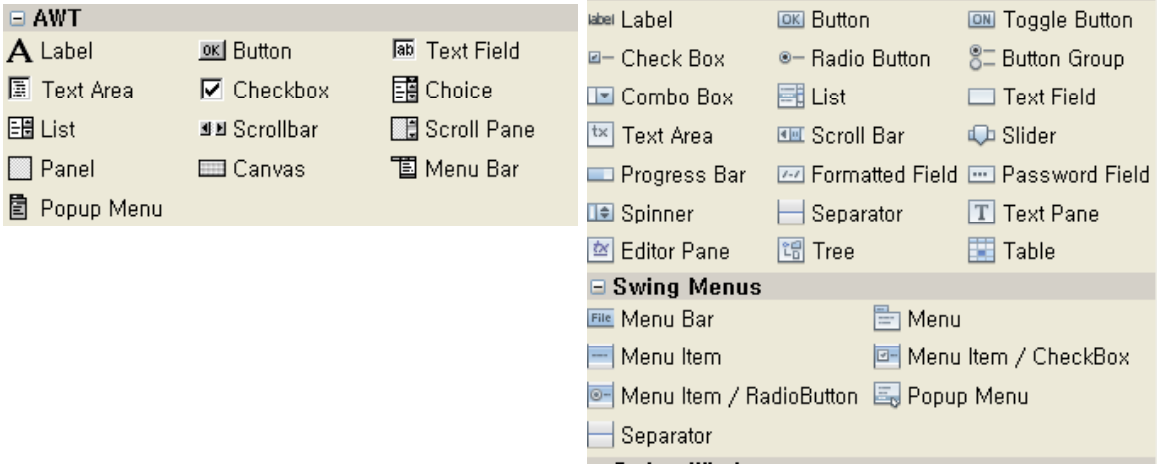


그림 17.1 그래픽 사용자 인터페이스는 컴포넌트들로 제작된다.

자바에서 GUI의 종류



AWT와 스윙



AWT와 SWING의 비교

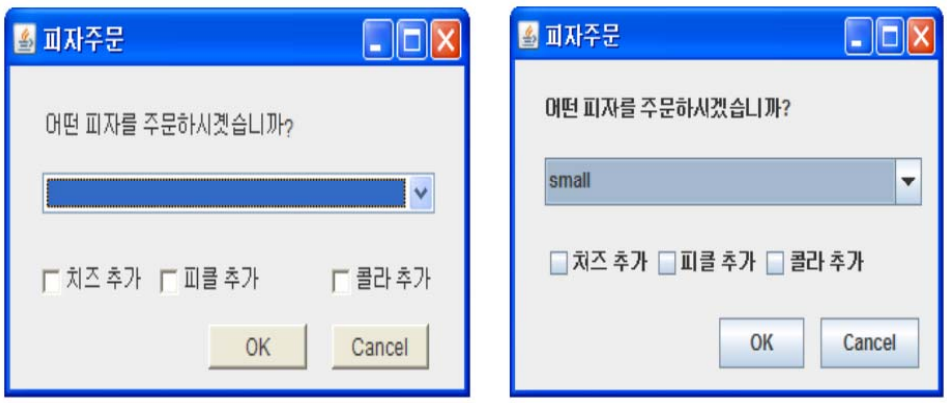


그림 17.3 동일한 GUI를 AWT와 스윙으로 만들어본 예

AWT와 스윙

표 19.1 AWT와 스윙의 비교

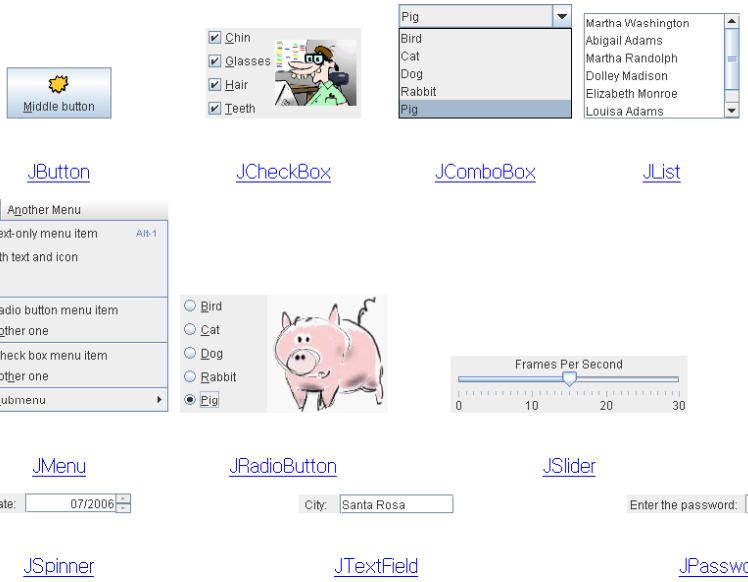
컴포넌트	AWT 버전	스윙 버전
버튼	Button	JButton
레이블	Label	JLabel
리스트	List	JList
...		
패스워드필드	없음	JPasswordField
슬라이더	없음	JSlider

중간 점검 문제

1. 그래픽 사용자 인터페이스를 구성하는 요소들을 무엇이라고 하는가?

2. AWT와 스윙의 차이점은 무엇인가?

스윙



스윙의 특징

- 스윙 GUI 컴포넌트
  - 형식화된 텍스트 입력이나 패스워드 필드 동작과 같은 복잡한 기능들이 제공된다.
- 자바 2D API
  - 그림이나 이미지, 애니메이션 기능을 제공한다.
  - 교체가능한 룩앤필(Look-and-Feel) 지원
- 데이터 전송
  - 자르기, 복사, 붙이기, 드래그앤 드롭 등의 데이터 전송 기능 제공
  - 되돌리기(undo)와 되풀이(redo) 기능을 손쉽게 제공

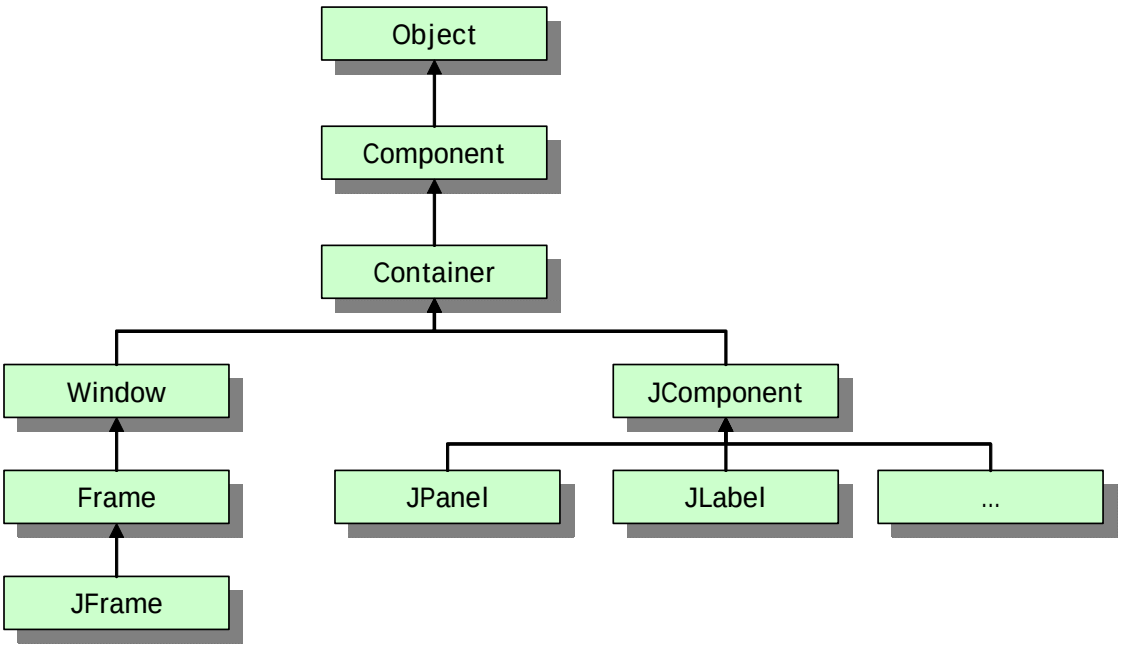
스윙 패키지

javax.accessibility	javax.swing.plaf	javax.swing.text
javax.swing	javax.swing.plaf.basic	javax.swing.text.html
javax.swing.border	javax.swing.plaf.metal	javax.swing.text.html.parser
javax.swing.colorchooser	javax.swing.plaf.multi	javax.swing.text.rtf
javax.swing.event	javax.swing.plaf.synth	javax.swing.tree
javax.swing.filechooser	javax.swing.table	javax.swing.undo

하지만 대부분의 프로그램은 스윙 API 중에서 아주 작은 부분 집합만을 사용한다. 따라서 대부분의 경우 다음의 두 가지의 패키지만을 포함하면 된다.

* javax.swing
* javax.swing.event

스윙의 클래스 구조



컨테이너와 컴포넌트

- *기본 컴포넌트*
  - JButton, JLabel, JCheckbox, JChoice, JList, JMenu, JTextField, JScrollbar, JTextArea, JCanvas 등이 여기에 속한다.
- *컨테이너 컴포넌트*
  - 다른 컴포넌트를 안에 포함할 수 있는 컴포넌트로서 JFrame, JDialog, JApplet, JPanel, JScrollPane 등이 여기에 속한다.

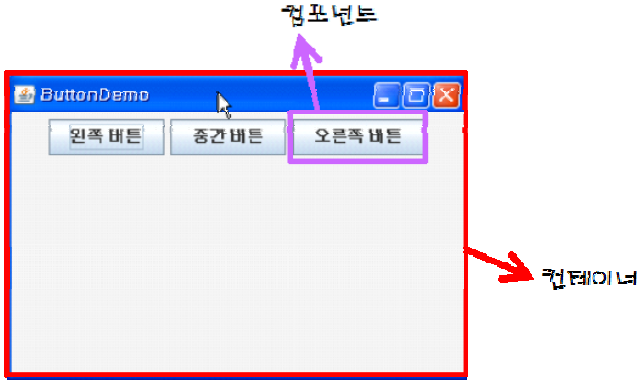


그림 17.6 컨테이너와 컴포넌트

- *1. 최상위 컨테이너*

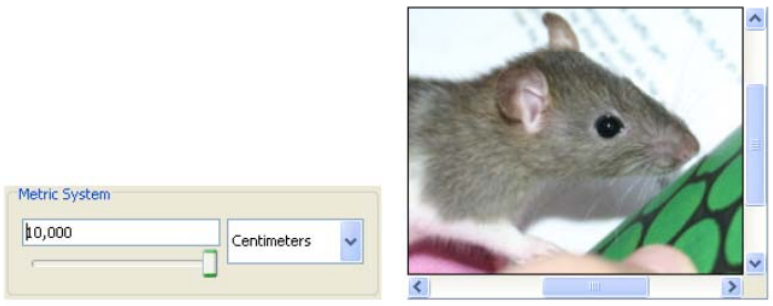
최상위 컨테이너란 절대 다른 컨테이너 안에 포함될 수 없는 컨테이너를 의미한다. 프레임(JFrame), 다이알로그(JDialog), 애플릿(JApplet) 등이 여기에 해당된다.

JApplet JDialog JFrame

그림 19.7 최상위 컨테이너(그림 출처: java.sun.com)

2. 일반 컨테이너

일반적인 컨테이너란 다른 컨테이너 안에 포함될 수 있는 컨테이너로 패널(JPanel), 스크롤페인(JScrollPane) 등을 의미한다.



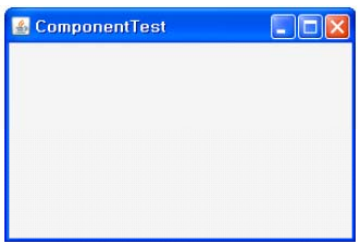
JPanel JScrollPane

중간 점검 문제

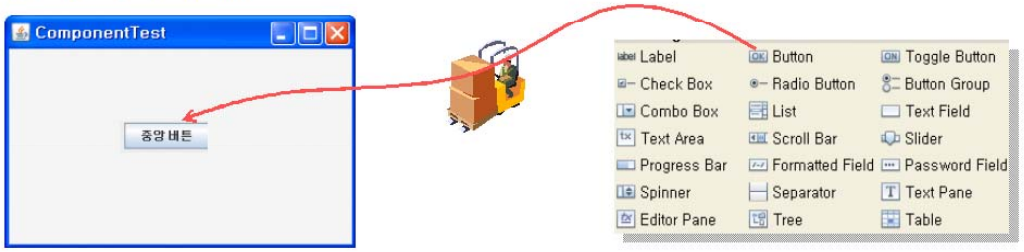
1. 컨테이너는 어떤 기능을 하는가?
2. 최상위 컨테이너의 특징은? 최상위 컨테이너에 속하는 클래스들을 나열하라.

GUI 작성 절차

(1) 컨테이너를 생성한다.



(2) 컴포넌트를 추가한다.



프레임 생성 #1

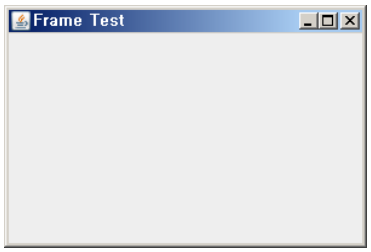
FrameTest.java

```
import javax.swing.*;

public class FrameTest {
        public static void main(String[] args) {
            JFrame f = new JFrame("Frame Test");
            f.setSize(300, 200);
            f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            f.setVisible(true);
        }

}
```

JFrame의 객체 생성



프레임 생성 #2

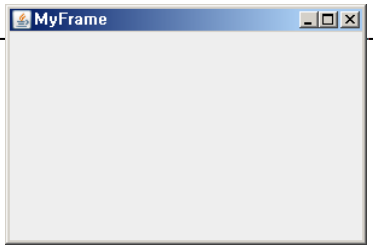
MyFrameTest.java

```
import javax.swing.*;

class MyFrame extends JFrame {
        public MyFrame() {
            setSize(300, 200);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setTitle("MyFrame");
            setVisible(true);
        }
}
public class MyFrameTest {
        public static void main(String[] args) {
            MyFrame f = new MyFrame();
        }
 }
```

Jframe을 상속하여서 MyFrame을 정의

MyFrame의 객체 생성



프레임 생성 #3

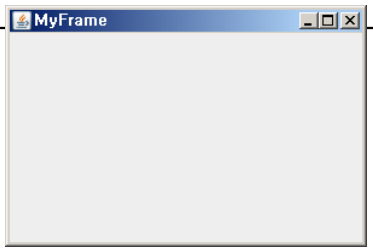
MyFrame.java

```
import javax.swing.*;

public class MyFrame extends JFrame {
        public MyFrame() {
                setSize(300, 200);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("MyFrame");
                setVisible(true);
        }
        public static void main(String[] args) {
                MyFrame f = new MyFrame();
        }
}
```

Jframe을 상속하여서 MyFrame을 정의

main()이 MyFrame 안으로 이동



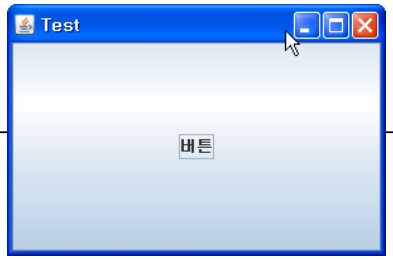
컴포넌트 생성과 추가

```
import javax.swing.*;

class MyFrame extends JFrame {
        public MyFrame() {
                JButton button = new JButton("버튼");
                this.add(button);
                setSize(300, 200);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("MyFrame");
                setVisible(true);
        }
}

public class MyFrameTest1 {
        public static void main(String[] args) {
                MyFrame f = new MyFrame();
        }
}
```

컴포넌트 생성 및 추가



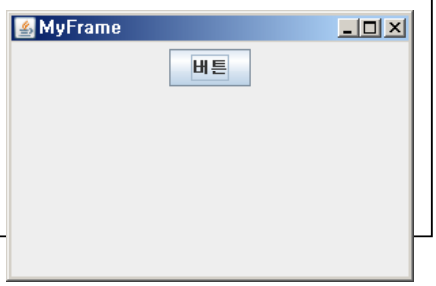
컴포넌트 생성과 추가

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

class MyFrame extends JFrame {
        public MyFrame() {
                JButton button = new JButton("버튼");
                this.add(button);
                setSize(300, 200);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("MyFrame");
                setLayout(new FlowLayout());
                setVisible(true);
        }
}

public class MyFrameTest2 {
        public static void main(String[] args) {
                MyFrame f = new MyFrame();
        }
}
```

배치 관리자 설정!



중간 점검 문제

1. 버튼이 있는 프레임을 생성하는 절차를 설명하라.
2. 프레임에 버튼을 두 개 추가하려면 어떻게 해야 하는가?

기초 컴포넌트

- 프레임(frame)
- 패널(panel)
- 레이블(label)
- 버튼(button)
- 텍스트 필드(text field)

프레임

- 메뉴를 붙일 수 있는 윈도우

생성자 또는 메소드	설명
void add(Component c)	지정된 컴포넌트를 프레임에 추가한다.
JMenuBar getJMenuBar()	이 프레임에 대한 메뉴를 얻는다.
void pack()	프레임을 크기를 추가된 컴포넌트들의 크기에 맞도록 조절한다.
void remove(Component c)	지정된 컴포넌트를 프레임에서 제거한다.
void setDefaultCloseOperation()	사용자가 프레임을 닫을 때 수행되는 동작을 설정한다. 일반적으로 JFrame.EXIT_ON_CLOSE으로 지정한다.
void setIconImage(Icon image)	프레임이 최소화되었을때의 아이콘 지정
void setLayout(LayoutManager layout)	프레임위에 놓이는 컴포넌트들을 배치하는 배치관리자 지정, 디폴트는 BorderLayout 배치관리자
void setLocation(int x, int y)	프레임의 x좌표와 y좌표를 지정한다.
void setResizeable(boolean value)	프레임의 크기 변경 허용 여부
void setSize(int width, int height)	프레임의 크기 설정
void setMenuBar(JMenuBar menu)	현재 프레임에 메뉴바를 붙인다.

패널

- 패널(panel)은 컴포넌트들을 가질 수 있는 컨테이너

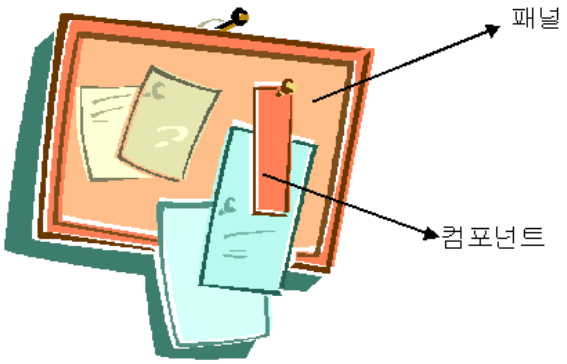


그림 19.12 패널은 컴포넌트를 붙일 수 있는 판이다.

패널의 메소드

- *생성자*

메소드	설명
JPanel()	새로운 패널을 생성한다.
JPanel(boolean isDoubleBuffered)	만약 매개변수가 참이면 더블 버퍼링을 사용한다.
JPanel(LayoutManager layout)	지정된 배치 관리자를 사용하는 패널을 생성한다.

- *메소드*

메소드	설명
void add(Component c)	지정된 컴포넌트를 패널에 추가한다.
void remove(Component c)	지정된 컴포넌트를 패널에서 제거한다.
void setLayout(LayoutManager layout)	배치 관리자를 지정한다. 디폴트는 FlowLayout이다.
void setLocation(int x, int y)	패널의 위치를 지정한다.
void setSize(int width, int height)	패널의 크기를 지정한다.
void setToolTipText(String text)	사용자가 마우스를 패널의 빈곳에 올려놓으면 툴팁을 표시한다.

레이블

- 레이블은 편집이 불가능한 텍스트를 표시.
  - (예) JLabel label = **new** JLabel("안녕하세요?");

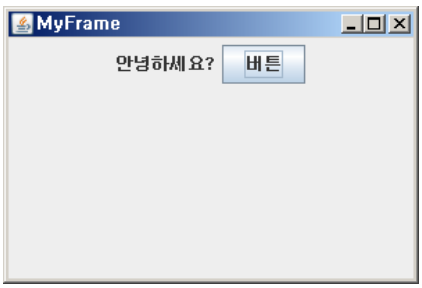
메소드	설명
String getText()	레이블의 텍스트를 반환한다.
void setText(String text)	레이블의 텍스트를 설정한다.
void setToolTipText(String text)	사용자가 마우스를 레이블 위에 올려놓으면 툴팁을 표시한다.
void setVisible(boolean value)	레이블을 보이게 하거나 감춘다.

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

class MyFrame extends JFrame {
        public MyFrame() {
                setSize(300, 200);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("MyFrame");

                JPanel panel = new JPanel();          // 패널 생성
                JLabel label = new JLabel("안녕하세요?");     // 레이블 생성
                JButton button = new JButton("버튼");         // 버튼 생성
                panel.add(label);                     // 패널에 레이블 추가
                panel.add(button);                    // 패널에 버튼 추가
                add(panel);                           // 패널을 프레임에 추가
                setVisible(true);
        }
}

public class MyFrameTest3 {
        public static void main(String[] args) {
                MyFrame f = new MyFrame();
        }
}
```



버튼

- 버튼은 사용자가 클릭했을 경우, 이벤트를 발생하여 원하는 동작을 하게 하는데 이용된다

메소드	설명
String getText()	버튼의 현재 텍스트를 반환한다.
void setText(String text)	버튼의 텍스트를 설정한다.
void doClick()	사용자가 버튼을 누른 것처럼 이벤트를 발생한다.
void setBorderPainted(booelan value)	버튼의 경계를 나타내거나 감춘다.
void setContentAreaFilled(boolean value)	버튼의 배경을 채울 것인지를 지정한다.
void setEnabled(boolean value)	버튼을 활성화하거나 비활성화한다.
void setRolloverEnabled(boolean value)	마우스가 버튼 위에 있으면 경계를 진하게 하는 롤오버 효과를 설정
void setToolTipText(String text)	사용자가 마우스를 버튼 위에 올려놓으면 툴팁을 표시한다.
void setVisible(boolean value)	버튼을 보이게 하거나 감춘다.

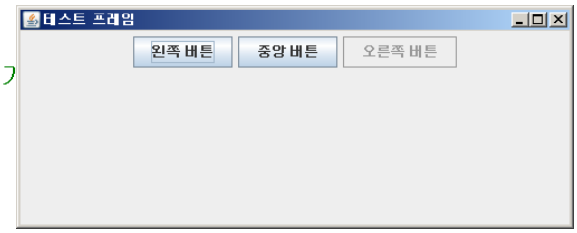
예제

```
class MyFrame extends JFrame {
        public MyFrame() {
          setSize(500, 200);
          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
          setTitle("테스트 프레임");

          JPanel panel = new JPanel();
          JButton b1 = new JButton();
          b1.setText("왼쪽 버튼");
          JButton b2 = new JButton("중앙 버튼");
          JButton b3 = new JButton("오른쪽 버튼");
          b3.setEnabled(false);      // 세 번째 버튼을 불활성으로 설정

          // 컴포넌트를 패널에 추가
          panel.add(b1);
          panel.add(b2);
          panel.add(b3);

          add(panel);       // 패널을 프레임에 추가
          setVisible(true);
        }
}
```



텍스트 필드

- 텍스트 필드(text field)는 입력이 가능한 한 줄의 텍스트 필드를 만드는 데 사용

- 생성자

생성자	설 명
JTextField()	TextField를 생성한다.
JTextField(int columns)	지정된 칸수를 가지고 있는 TextField를 생성한다.
JTextField(String text)	지정된 문자열로 초기화된 TextField를 생성한다.

- 메소드

메소드	설 명
void setText(String text)	지정된 문자열을 텍스트 필드에 쓴다.
String getText()	텍스트 필드에 입력된 문자열을 반환한다.
void setEditable(boolean) boolean isEditable()	사용자가 텍스트를 입력할 수 있는지 없는지를 설정하고 반환한다.

예제

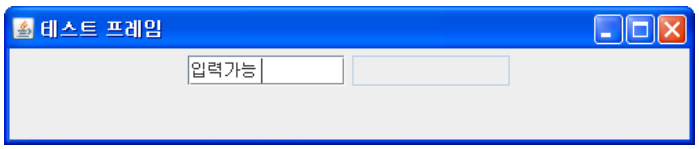
```
class MyFrame extends JFrame {
        public MyFrame() {
                setSize(500, 100);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("테스트 프레임");

                JPanel panel = new JPanel();
                JTextField t1 = new JTextField(10);
                JTextField t2 = new JTextField(10);
                t2.setEditable(false);

                // 컴포넌트를 패널에 추가
                panel.add(t1);
                panel.add(t2);

                add(panel); // 패널을 프레임에 추가
                setVisible(true);
        }
}
```

텍스트 필드 생성



Q & A