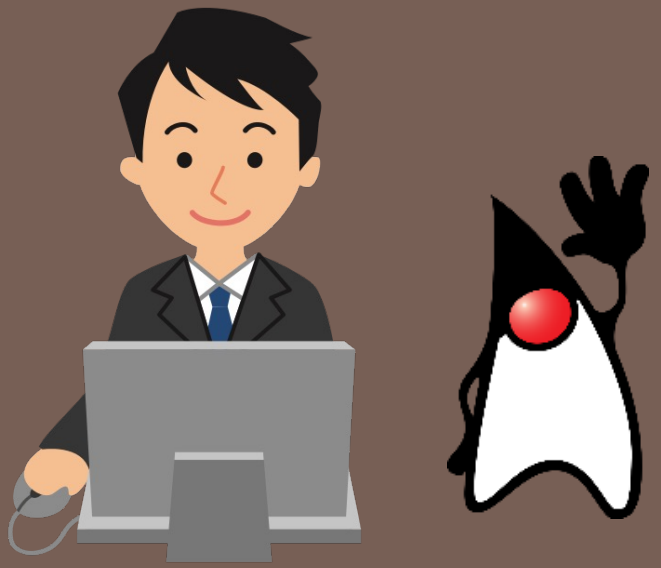
파워자바(개정3판)



9장 자바GUI 기초

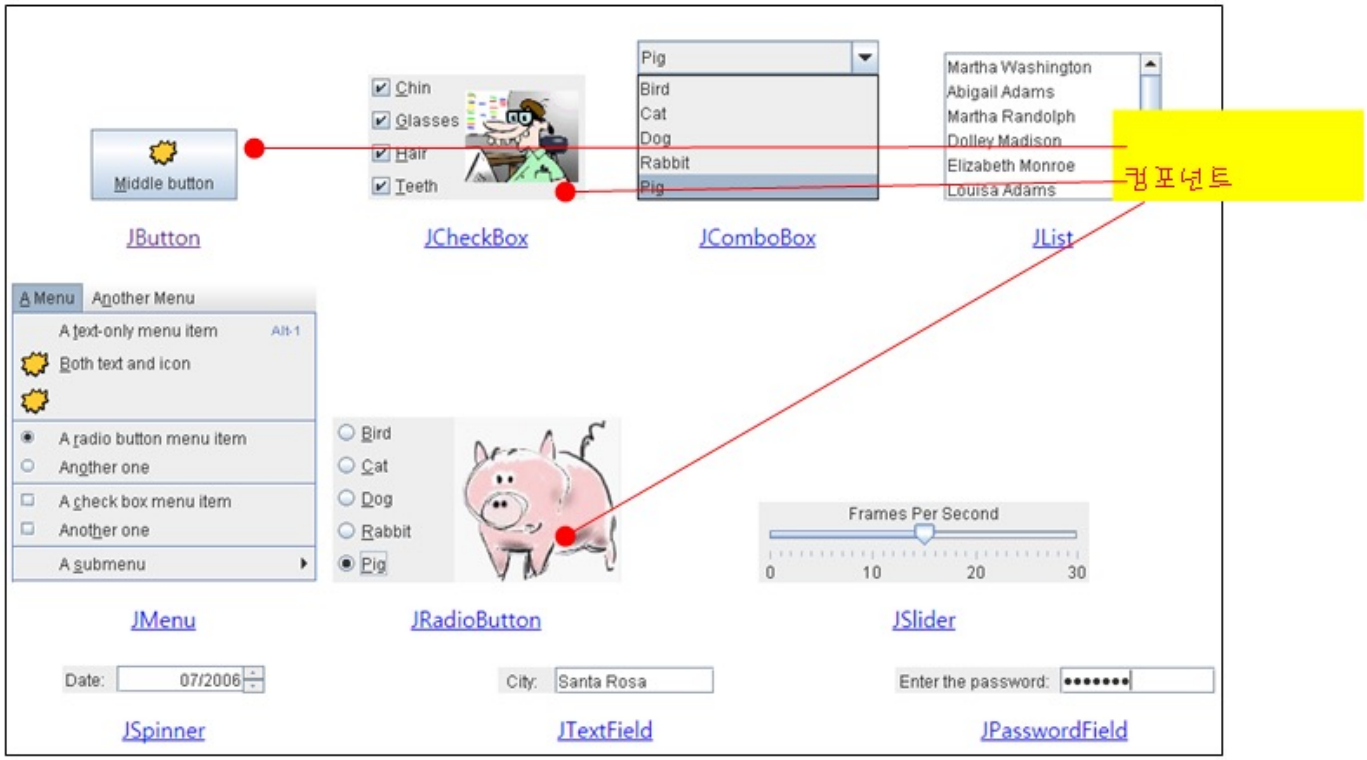
9장의 목표

1. 객체 지향 기법을 사용하여 GUI 화면을 구성할 수 있나요?
2. 버튼, 텍스트 필드, 레이블을, 원하는 대로 화면에 배치할 수 있나요?
3. 패스워드가 보이지 않게 입력받을 수 있나요?
4. 화면에 이미지를 표시할 수 있나요?

Java Swing

그래픽 사용자 인터페이스

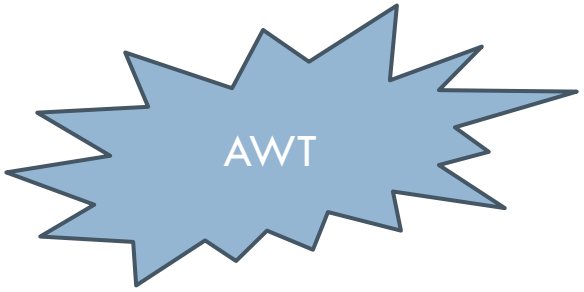
- 그래픽 사용자 인터페이스(Graphical User Interface, 간단히 GUI)는 컴포넌트들로 구성된다.

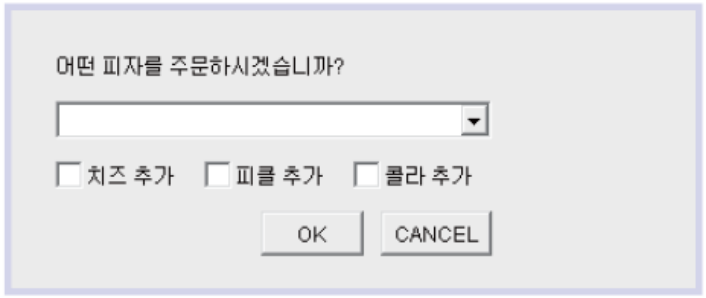


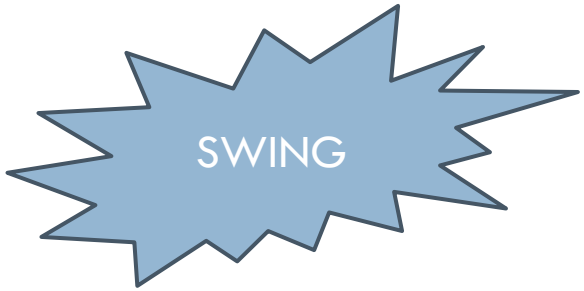
자바에서 GUI의 종류

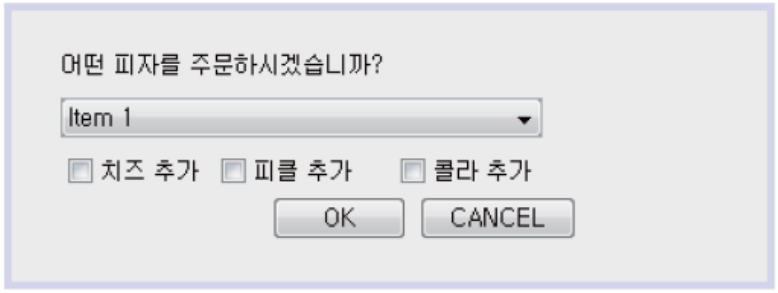
Java AWT	Java Swing
AWT는 플랫폼에 의존적이다.	스윙은 플랫폼에 독립적이다.
AWT 컴포넌트는 용량이 크다.	스윙 컴포넌트는 용량이 가볍다.
AWT 교체할 수 있는 **룩앤필(look and feel)을 지원하지 않는다**.	스윙은 교체할 수 있는 **룩앤필(look and feel)을 지원한다**.
컴포넌트의 개수가 적다.	컴포넌트의 개수가 많다.

AWT와 SWING의 비교



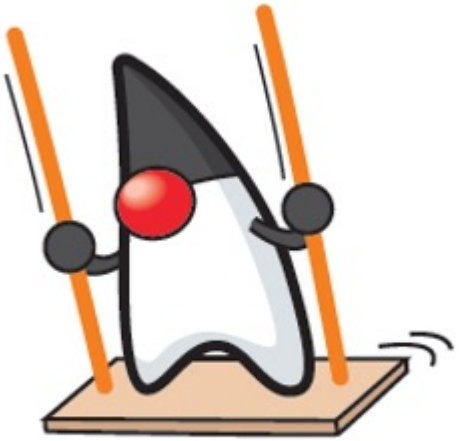






스윙 패키지

- java.awt — GUI 컴포넌트를 위한 부모 클래스들을 제공하고 추가로 Color나 Point와 같은 유틸리티 타입의 클래스들을 포함하고 있다.
- java.awt.event — GUI 컴포넌트로부터 발생되는 이벤트(예를 들면 버튼 클릭 이벤트)를 처리하기 위한 클래스와 인터페이스를 가지고 있다.
- javax.swing — 버튼이나 텍스트 필드, 프레임, 패널과 같은 GUI 컴포넌트들을 가지고 있다.



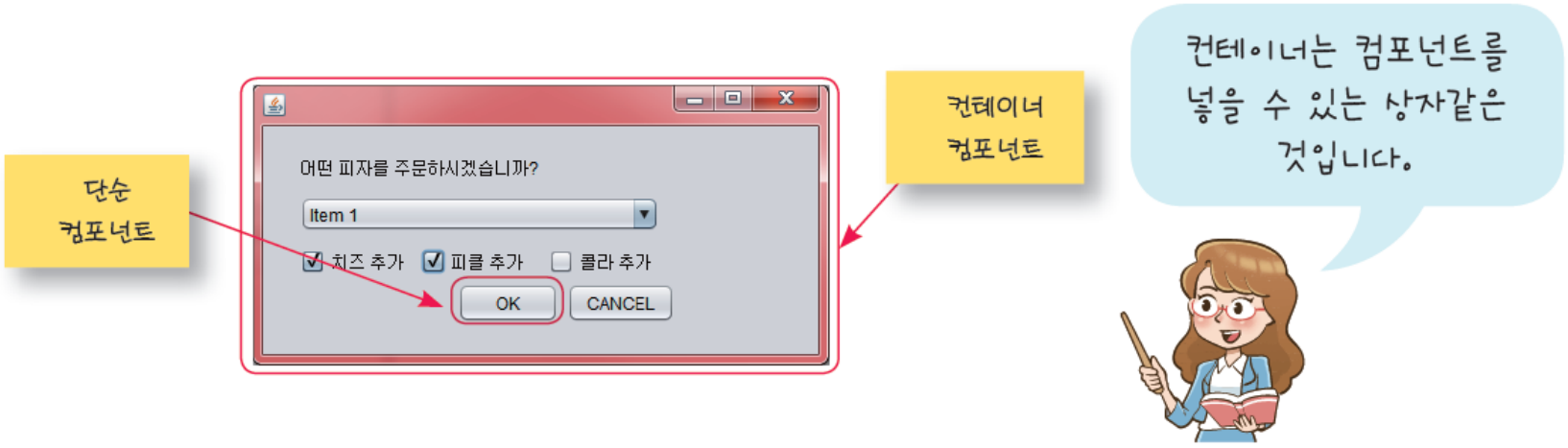
Java Swing

중간점검

1. 운영체제에 따라서 GUI 컴포넌트의 모양이 달라지는 것은 AWT인가? 스윙인가?
2. 순수 자바로 작성된 GUI는 AWT인가? 스윙인가?
3. GUI를 구성하는 객체들을 무엇이라고 부르는가?

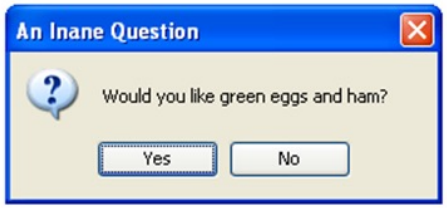
컨테이너와 컴포넌트

- *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.



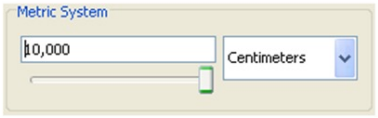
컨테이너의 종류

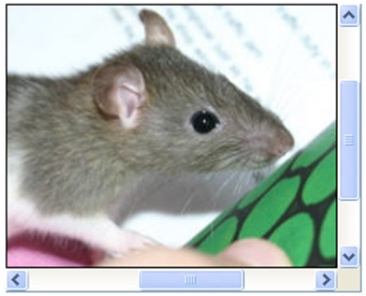
- *최상위 컨테이너:* 절대 다른 컨테이너 안에 포함될 수 없는 컨테이너를 의미한다. 프레임(JFrame), 다이알로그(JDialog), 애플릿(JApplet) 등이 여기에 해당된다.



JApplet JDialog JFrame

- *일반 컨테이너:* 다른 컨테이너 안에 포함될 수 있는 컨테이너로 패널(JPanel), 스크롤 페인(JScrollPane) 등을 의미한다.

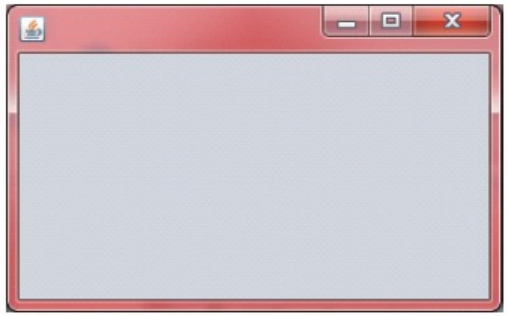




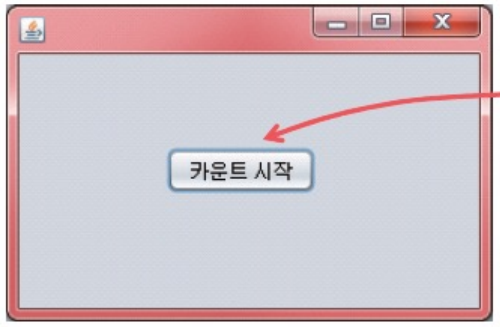
JPanel JScrollPane

GUI 작성 절차

(1) 컨테이너를 생성한다.

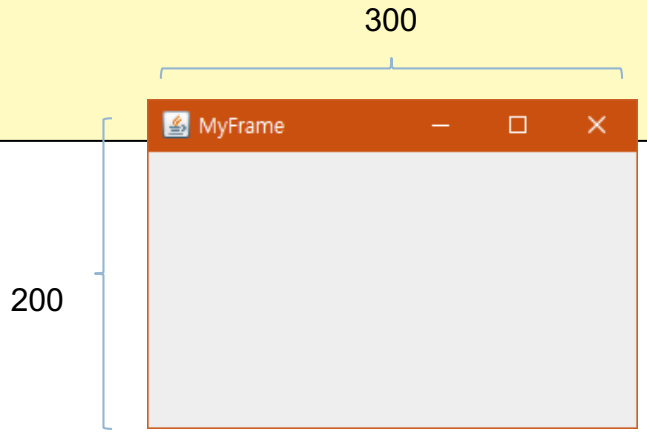


(2) 컴포넌트를 추가한다.



프레임을 생성하는 방법 #1

```
import javax.swing.*;                                          // (1)

public class FrameTest {
      public static void main(String[] args) {


          JFrame f = new JFrame("Frame Test"); // (2)
          f.setTitle("MyFrame");                               // (3)
          f.setSize(300, 200);                                          // (4)
          f.setVisible(true);                                           // (5)
          f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);     // (6)
      }
}
```



참고

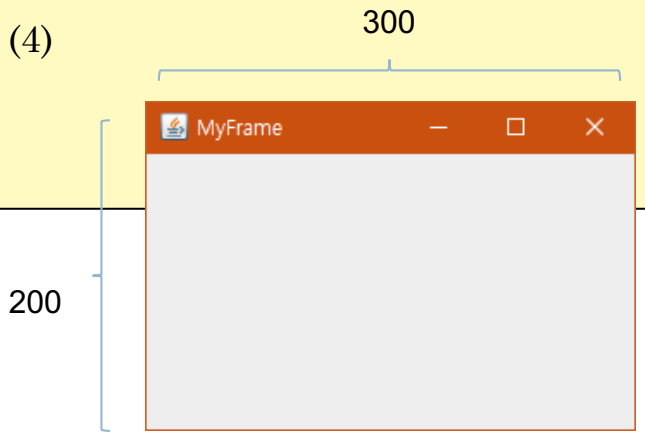
main()에서는 MyFrame 객체를 생성한 후에 종료한다. 하지만 위의 프로그램을 실행시켜 보면 윈도우는 종료되지 않는다. 어떻게 된 일일까?


```
public static void main(String[] args) {
   MyFrame f = new MyFrame();      // 이 문장이 종료되면 프로그램도 종료?
}
```

자바 스윙에서 JFrame 객체를 만들면 윈도우를 담당하는 새로운 스레드가 하나 생성된다(스레드는 독립적으로 실행되는 코드이다). 따라서 main()이 종료되더라도 윈도우는 없어지지 않는다.

프레임을 생성하는 방법 #2

```
import javax.swing.*;                                          // (1)

public class MyFrame extends JFrame {   // (2)
      public MyFrame() {                                                  // (3)
            setSize(300, 200);
            setTitle("MyFrame");


            setVisible(true);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      }
      public static void main(String[] args) {
            MyFrame f = new MyFrame();                  // (4)
      }
}
```

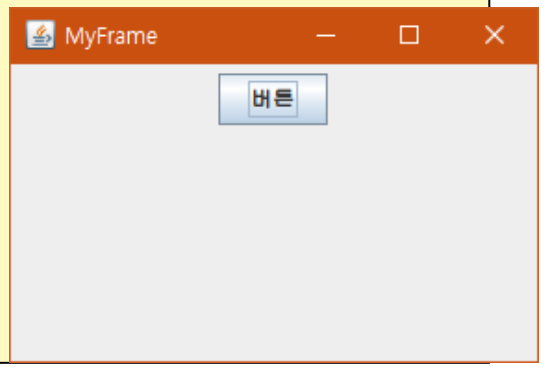


참고

앞의 프로그램에서 main() 메소드가 MyFrame 클래스 안에 정의되었다. main()은 어떤 클래스 안에서도 선언될 수 있다. main() 앞에서는 static가 붙어 있다. 즉 정적 메소드라는 이야기다. 정적 메소드는 객체를 생성하지 않아도 얼마든지 호출할 수 있다. main()은 외부에서 객체를 생성하지 않고도 호출할 수 있어야 되기 때문이다. 클래스 안에 main()이 정의되어 있으면 그 클래스를 독립적으로 실행할 수 있다. 새로운 클래스를 작성하였을 때 테스트하고 싶으면 클래스 안에 main()을 만들면 된다. 그리고 그 클래스를 실행시키면 된다. 이클립스에서는 클래스 위에서 마우스 오른쪽 버튼을 누르고 "Run As Java Application" 메뉴를 선택한다.

프레임에 버튼 추가하기

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


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

		setLayout(new FlowLayout());                    // (1)
		JButton button = new JButton("버튼");           // (2)
		add(button);                                    // (3)
		setVisible(true);
		setDefaultCloseOperation(EXIT_ON_CLOSE);
	}
	public static void main(String[] args) {
		MyFrame f = new MyFrame();
	}
}
```



경고

만약 배치 관리자를 FlowLayout으로 지정하지 않고 버튼을 프레임에 추가하면 버튼이 전체 화면을 차지하게 된다. 이유는 컴포넌트를 배치하는 배치 관리자가 BorderLayout이기 때문이다. 이번 장 뒷부분에서 학습한다.

참고

참고

setVisible()의 위치

setVisible(true) 문장의 위치에 주의하자. add() 문장의 앞에서 setVisible(true) 문장이 실행되면 화면에 아무것도 나오지 않을 수 있다. 항상 모든 자식 컴포넌트들을 컨테이너에 추가한 후에 맨 마지막에 setVisible(true) 문장을 실행하도록 한다.

중간점검

중간점검

1. 최상위 컨테이너 중에서 하나만 말해보자.
2. 버튼이 하나 있는 GUI를 작성하는 절차를 말해보자.

컨테이너 살펴보기: JFrame

- JFrame은 아래 그림과 같이 수많은 조상 클래스들을 가지고 있다. 우리가 상속에서 살펴보았듯이 조상 클래스가 제공하는 속성과 메소드들은 자식 클래스가 사용할 수 있다. 따라서 JFrame 클래스의 조상 클래스가 가지고 있는 속성과 메소드들도 모두 사용이 가능하다.

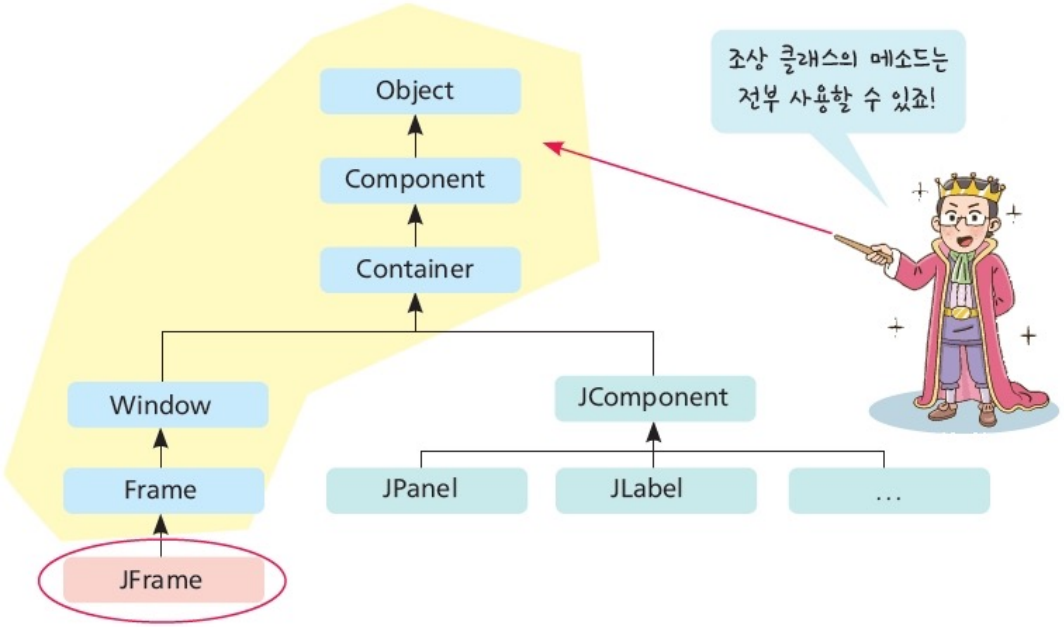
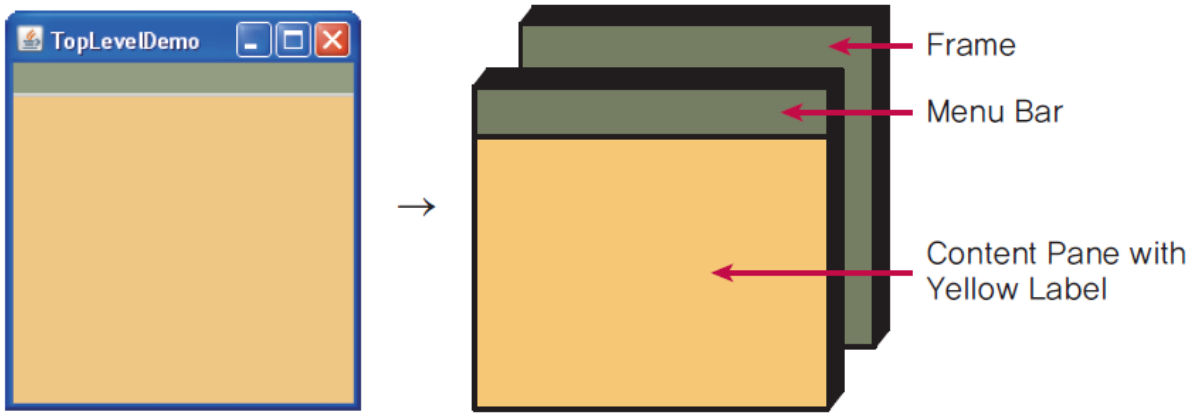


그림 9.6 스윙 관련 클래스의 계층 구조

JFrame 클래스

- 컨테이너는 컴포넌트들을 트리(tree) 형태로 저장한다. 최상위 컨테이너는 이 트리의 루트 노드가 된다.
- 최상위 컨테이너는 내부에 콘텐트 페인(content pane)을 가지고 있다. 여기에 화면에 보이는 컴포넌트를 저장한다.
- 최상위 컨테이너에는 메뉴바를 추가할 수 있다.



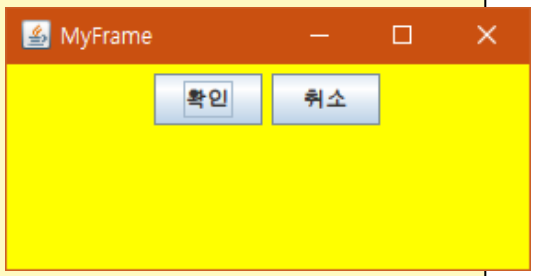
중요한 메소드

- add(component) - 프레임에 컴포넌트를 추가한다.
- setLocation(x, y) , setSize(width, height) - 프레임의 위치와 크기를 설정한다.
- setIconImage(IconImage) - 윈도우 시스템에 타이틀 바, 태스크 스위처에 표시할 아이콘을 알려준다.
- setTitle() - 타이틀 바의 제목을 변경한다.
- setResizable(boolean) - 사용자가 크기를 조절할 수 있는지를 설정한다.

예제: JFrame의 메소드 사용 예제

- JFrame 컨테이너의 배경색을 노랑색으로 변경하고 버튼을 2개 추가하는 프로그램은 다음과 같다.

```
public class MyFrame extends JFrame {
        public MyFrame() {
                setSize(300, 150);                  // (1) JFrame의 크기를 설정한다.
                setLocation(200, 300);              // (2) JFrame의 위치를 설정한다.
                setTitle("MyFrame");
                setLayout(new FlowLayout());
                getContentPane().setBackground(Color.yellow); // (3) 배경색을 변경한다.
                JButton button1 = new JButton("확인");


                JButton button2 = new JButton("취소");
                this.add(button1);
                this.add(button2);
                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
        }
   public static void main(String[] args) {
      MyFrame f = new MyFrame();
  }
}
```



JPanel 클래스

- 패널(panel)은 컴포넌트들을 부착할 수 있도록 설계된 컨테이너 중의 하나이다(최상위 컨테이너는 아니다).
- 별도의 패널을 쓰는 것이 유지 보수 및 배치 관리에 유리한 경우가 많다. 예를 들어서 프레임에 2장의 패널을 부착하고 각 패널의 배경색을 다르게 할 수 있다

그림 9.8 패널은 컴포넌트를 붙일 수 있는 판이다.

JPanel 클래스

- add(aComponent): 패널에 컴포넌트를 추가한다.
- remove(aComponent): 패널에 컴포넌트를 삭제한다.
- setBackground(Color c): 패널의 배경색을 변경한다.

그림 9.8 패널은 컴포넌트를 붙일 수 있는 판이다.

예제: 패널 사용하기

```
public class MyFrame extends JFrame {
        public MyFrame() {
                JPanel panel = new JPanel(); // 패널을 생성한다.
                panel.setBackground(Color.orange); // 패널의 배경색을 변경한다.

                JButton b1 = new JButton("Button 1"); // 버튼을 생성한다.
                b1.setBackground(Color.yellow); // 버튼의 배경색을 변경한다.

                JButton b2 = new JButton("Button 2");
                b2.setBackground(Color.green);



                panel.add(b1); // 버튼을 패널에 추가한다.
                panel.add(b2); // 버튼을 패널에 추가한다.
                add(panel); // 패널을 프레임에 추가한다.
                setSize(300, 150);
                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
        }

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

중간점검

1. 컨테이너와 단순 컴포넌트가 다른 점은 무엇인가?
2. 프레임에서 컴포넌트를 붙일 수 있는 영역을 무엇이라고 하는가?
3. 패널은 최상위 컨테이너인가?

배치 관리자(layout manager)

- 컨테이너 안의 각 컴포넌트의 위치와 크기를 결정하는 작업

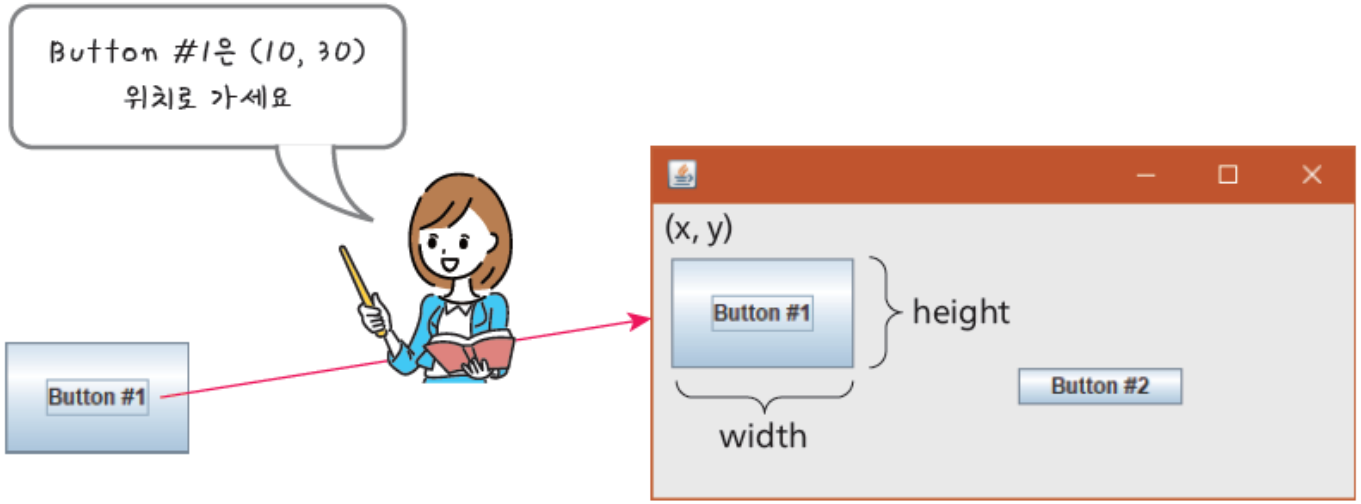
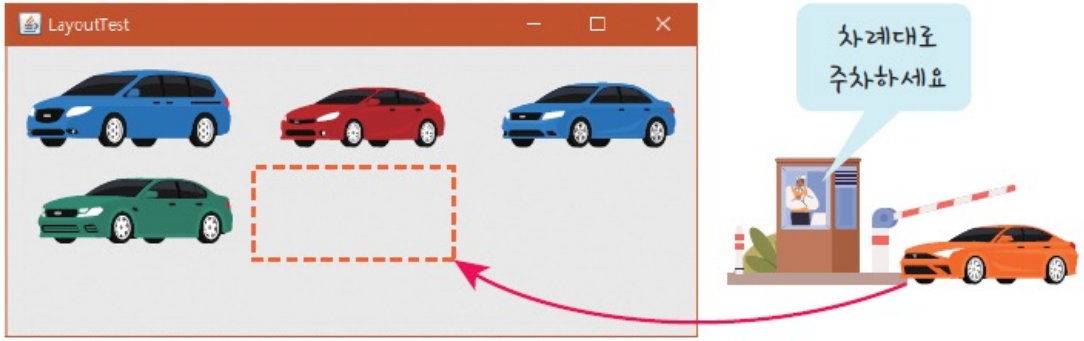


그림 9.9 배치 관리자의 개념

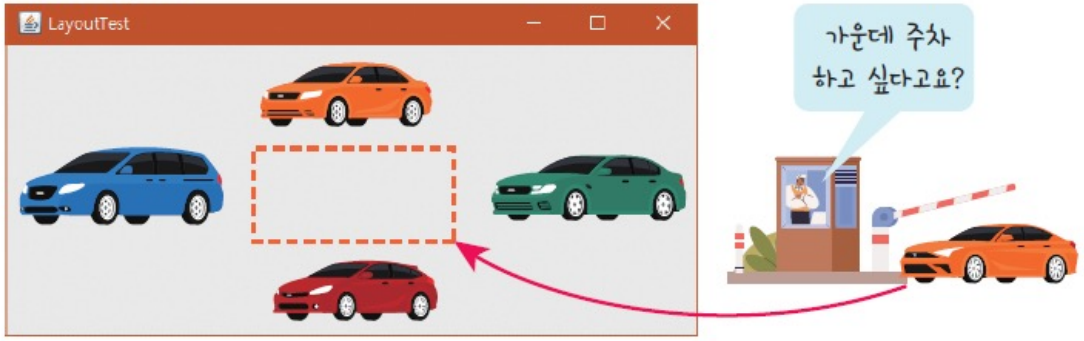
배치관리자의 종류

FlowLayout



컨테이너에 추가되는 순서대로 컴포넌트를 부착한다. 위쪽에서 아래쪽으로, 왼쪽에서 오른쪽으로 배치한다. 패널의 기본 배치 관리자이다.

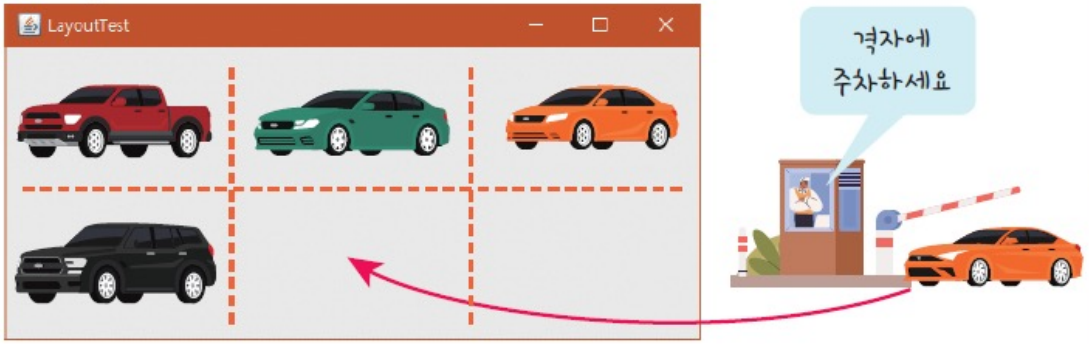
BorderLayout



컨테이너의 영역을 동서남북, 중앙의 5개의 영역으로 구분하여 이 영역에 컴포넌트를 배치한다. 프레임의 기본 배치 관리자이다.

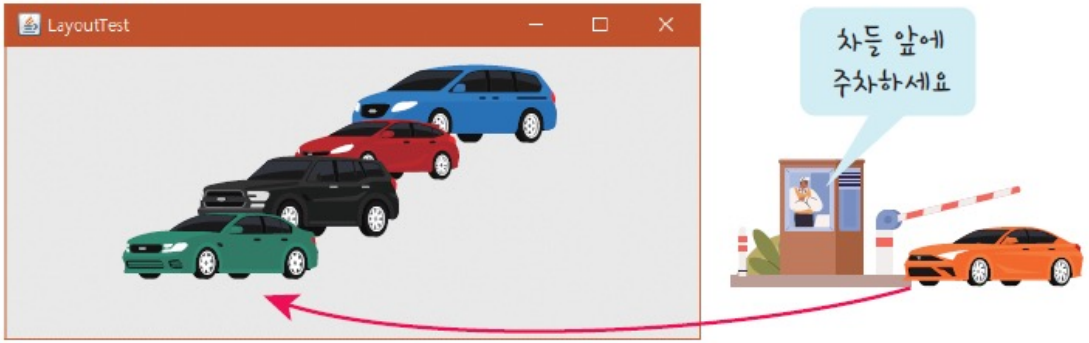
배치관리자의 종류

GridLayout



컨테이너의 공간을 동일한 크기의 격자로 나누고 이 격자에 컴포넌트를 배치한다.

CardLayout



컨테이너에 컴포넌트를 카드처럼 겹치게 쌓아서 배치한다.

배치 관리자의 설정

1. 생성자를 이용하는 방법

```
JPanel panel = new JPanel(new BorderLayout());
```

2. setLayout() 메소드 이용

```
panel.setLayout(new FlowLayout());
```

FlowLayout

- 컴포넌트들을 왼쪽에서 오른쪽으로 버튼을 배치한다.
- 패널과 애플릿의 디폴트 배치 관리자이다.

- FlowLayout()
- FlowLayout(int align) // align은 정렬 방법을 지정한다.
- FlowLayout(int align, int hGap, int vGap) // 간격을 지정한다.

FlowLayout

```
public class MyFrame extends JFrame {
        public MyFrame() {
                setTitle("FlowLayoutTest");
                setSize(300, 150);


                setLayout(new FlowLayout());

                add(new JButton("Button1"));
                add(new JButton("Button2"));
                add(new JButton("Button3"));
                add(new JButton("Button4"));
                add(new JButton("Button5"));
                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
        }

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

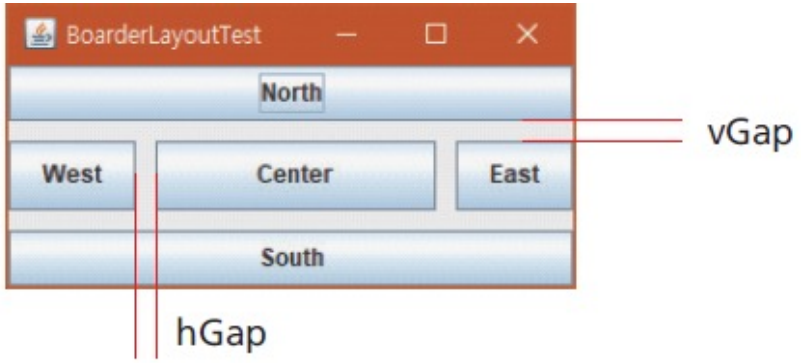
BorderLayout

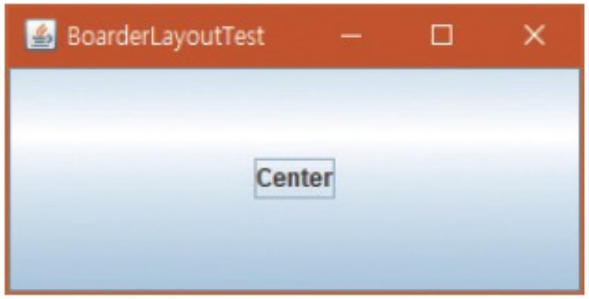
- 컴포넌트들이 5개의 영역인 North(상), South(하), East(좌측), West(우측), Center(중앙)중 하나로 추가된다.

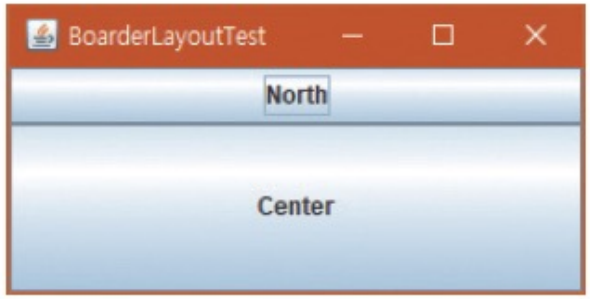

```
add(button, "South");
```

BorderLayout

- BorderLayout()
- BorderLayout(int hGap, int vGap)

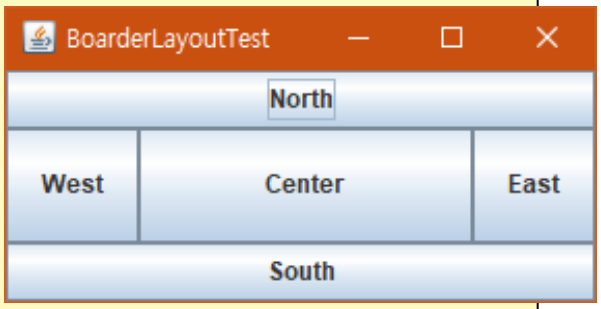






예제: BorderLayout

```
public class MyFrame extends JFrame {
        public MyFrame() {
                setTitle("BoarderLayoutTest");
                setSize(300, 150);
                setLayout(new BorderLayout()); // (1)



                JButton b1 = new JButton("North");
                JButton b2 = new JButton("South");
                JButton b3 = new JButton("East");
                JButton b4 = new JButton("West");
                JButton b5 = new JButton("Center");

                add(b1, "North");
                add(b2, "South");
                add(b3, "East");
                add(b4, "West");
                add(b5, "Center");

                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
        }
```



…

GridLayout

- GridLayout은 컴포넌트를 격자 모습으로 배치한다.

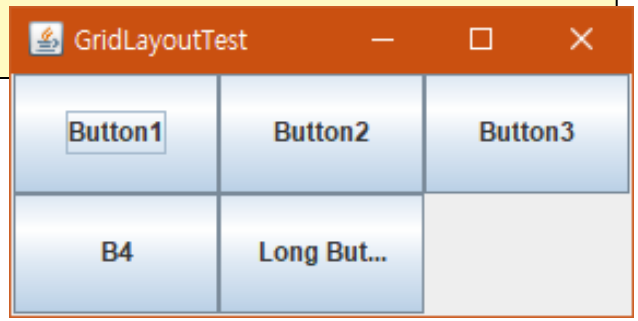
GridLayout

- GridLayout은 컴포넌트를 격자 모습으로 배치한다.

  - GridLayout() # 1행과 1열의 격자
  - GridLayout(int rows, int cols) # rows 행과 cols 열
  - GridLayout(int rows, int cols, int hGap, int vGap) # 간격 지정

GridLayout

```
public class MyFrame extends JFrame {
        public MyFrame() {
                setTitle("GridLayoutTest");
                setSize(300, 150);
                setLayout(new GridLayout(2, 3)); //



                add(new JButton("Button1"));
                add(new JButton("Button2"));
                add(new JButton("Button3"));
                add(new JButton("B4"));
                add(new JButton("Long Button5"));
                setVisible(true);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
...
```

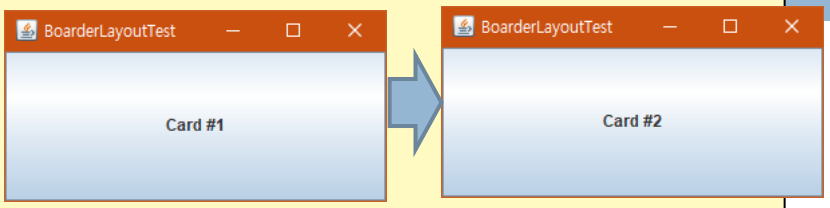


CardLayout

- CardLayout은 한 번에 하나의 컴포넌트만 볼 수 있게 배치하는 관리자이다.
  - next(container): 주어진 컨테이너의 다음 카드로 이동한다.
  - previous(container): 주어진 컨테이너의 이전 카드로 이동한다.
  - first(container): 주어진 컨테이너의 첫 번째 카드로 이동한다.
  - last(container): 주어진 컨테이너의 마지막 카드로 이동한다.

CardLayout

```
public class MyFrame extends JFrame {
        JButton b1, b2, b3;
        Container cPane;
        CardLayout layoutm;

        public MyFrame() {
                setTitle("BoarderLayoutTest");
                setSize(300, 150);
                cPane = getContentPane();
                layoutm = new CardLayout();
                setLayout(layoutm); // (1)

                JButton b1 = new JButton("Card #1");
                JButton b2 = new JButton("Card #2");
                JButton b3 = new JButton("Card #3");

                add(b1);
                add(b2);
                add(b3);
                b1.addActionListener(e->layoutm.next(cPane));
                b2.addActionListener(e->layoutm.next(cPane));
                b3.addActionListener(e->layoutm.next(cPane));
                setVisible(true);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
        }
        ...}
```



절대 위치로 배치

1. 배치 관리자를 null로 설정한다.

```
setlayout(null);
```

2. add() 메소드를 사용하여 컴포넌트를 컨테이너에 추가한다.

```
Button b = Button("Button #1");
add(b);
```

3. setSize(w, h)와 setLocation(x, y)을 사용하여 컴포넌트의 위치와 크기를 지정한다. 아니면 setBounds(x, y, w, h)를 사용하여 위치와 크기를 동시에 지정해도 된다.

```
b.setBounds(x, y, w, h);
```

절대 위치로 배치

```
public class MyFrame extends JFrame {
        private JButton b1, b2;

        public MyFrame() {
                setTitle("Absolute Position Test");
                setSize(300, 150);
                setLayout(null); // (1)

                b1 = new JButton("Button #1");
                add(b1); // (2)
                b1.setLocation(50, 30); // (3)
                b1.setSize(90, 50);
                b2 = new JButton("Button #2");
                add(b2);
                b2.setBounds(180, 30, 90, 20);
                setVisible(true);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        }
        public static void main(String args[]) {
                MyFrame f = new MyFrame();
        }
}
```

패널의 배치 관리자를 지정하지 않는다. 즉 절대 위치를 사용하겠다는 의미이다.

각 버튼의 크기와 위치를 setSize(), setLocation() 메소드를 이용하여 지정한다. setBounds(x, y, width, height)를 사용하여서 위치와 크기를 동시에 지정하여도 된다.

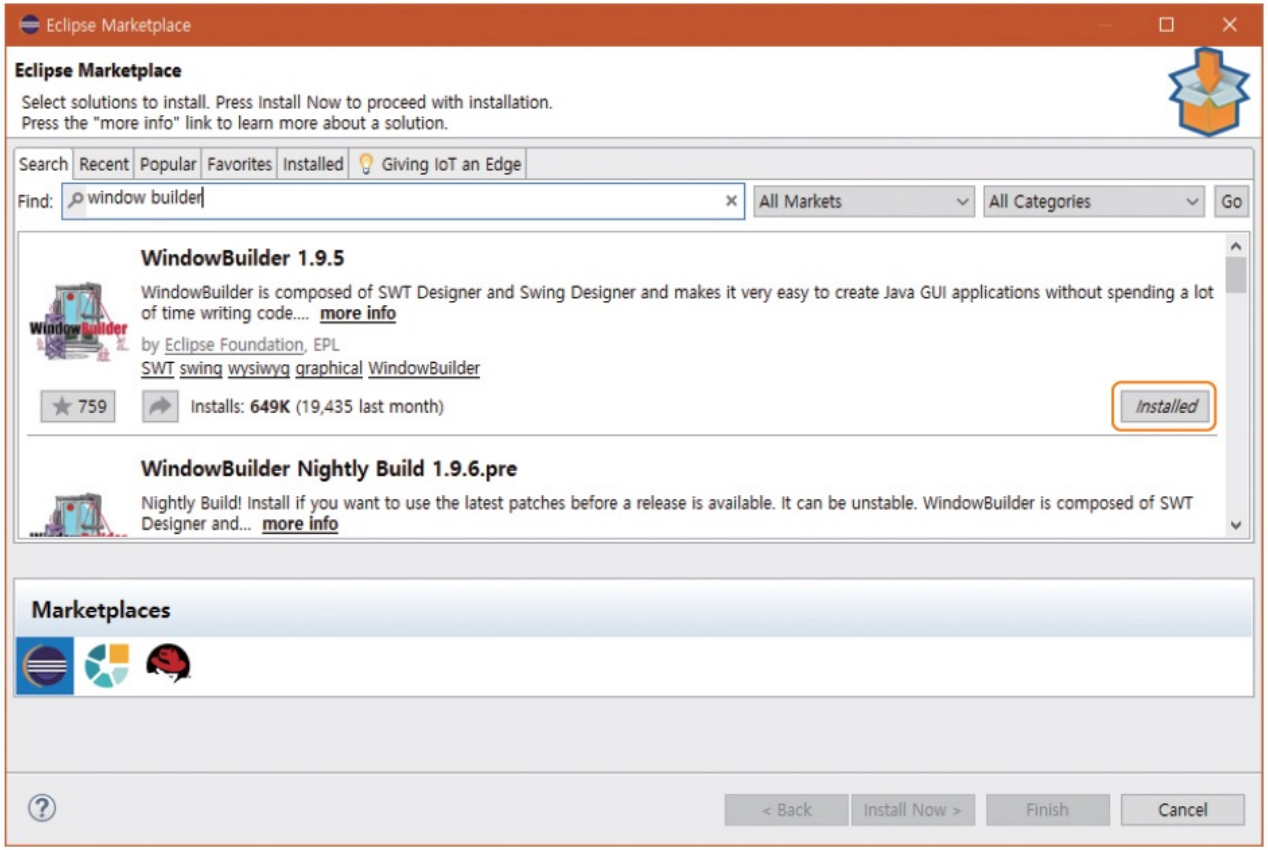
중간점검

중간점검

1. 격자처럼 컴포넌트를 배치하는 배치 관리자 클래스는?
2. 순차적으로 컴포넌트를 배치하는 배치 관리자 클래스는?
3. 동서남북으로 컴포넌트를 배치하는 배치 관리자 클래스는? 컴포넌트 사이의 수평, 수직 간격을 10 픽셀, 20 픽셀로 하는 GridLayout 배치 관리자를 생성해보자.
4. 컨테이너의 배치 관리자를 제거하려면 어떻게 하면 되는가?

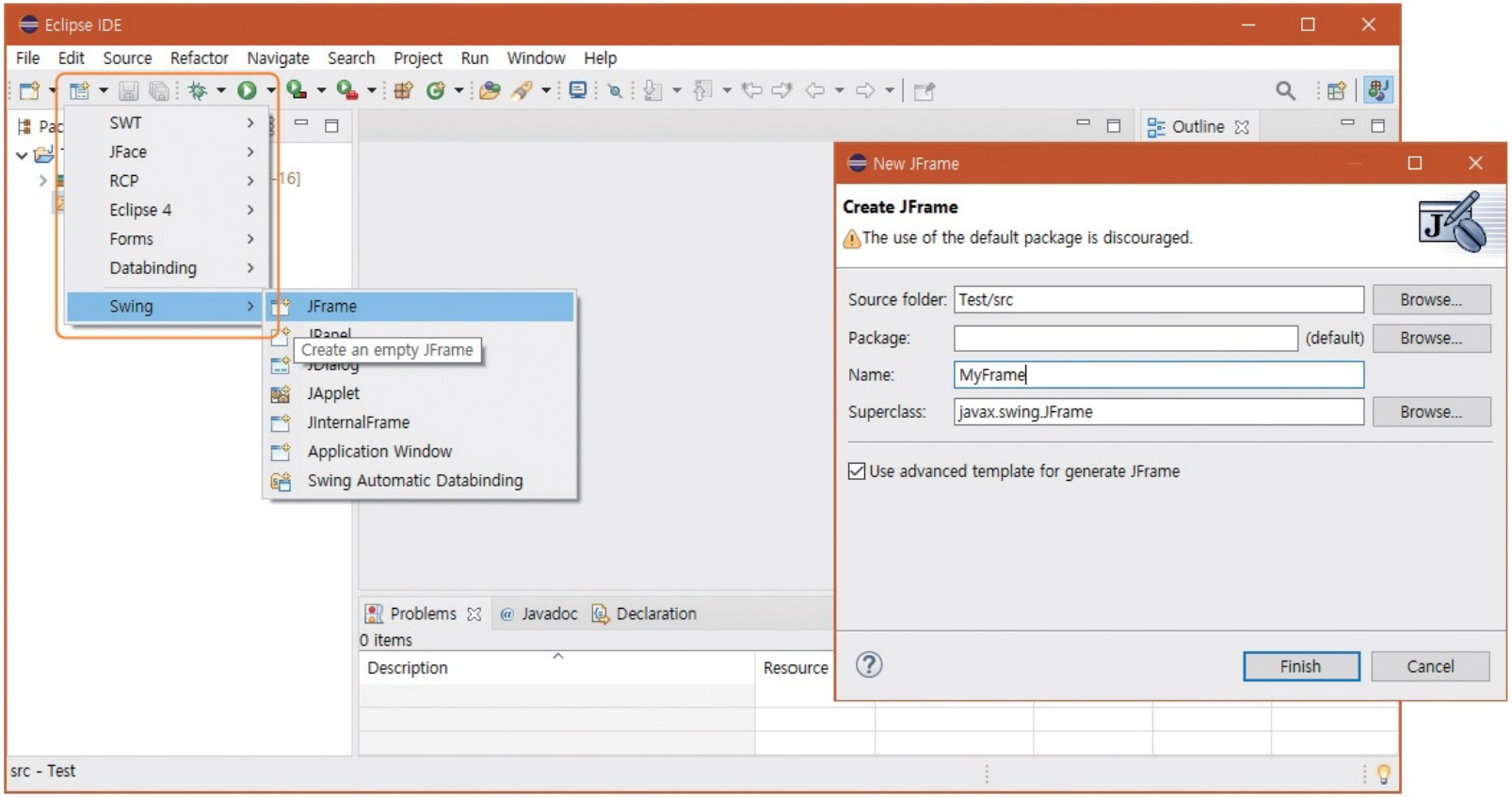
스윙 비주얼 디자이너: WindowBuilder

- 이클립스의 [Help] -> [Eclipe Marketplace]를 선택한 후 “WindowBuilder”를 검색하고 [Install] 버튼을 누른다. 설치가 종료되면 이클립스를 다시 시작한다.



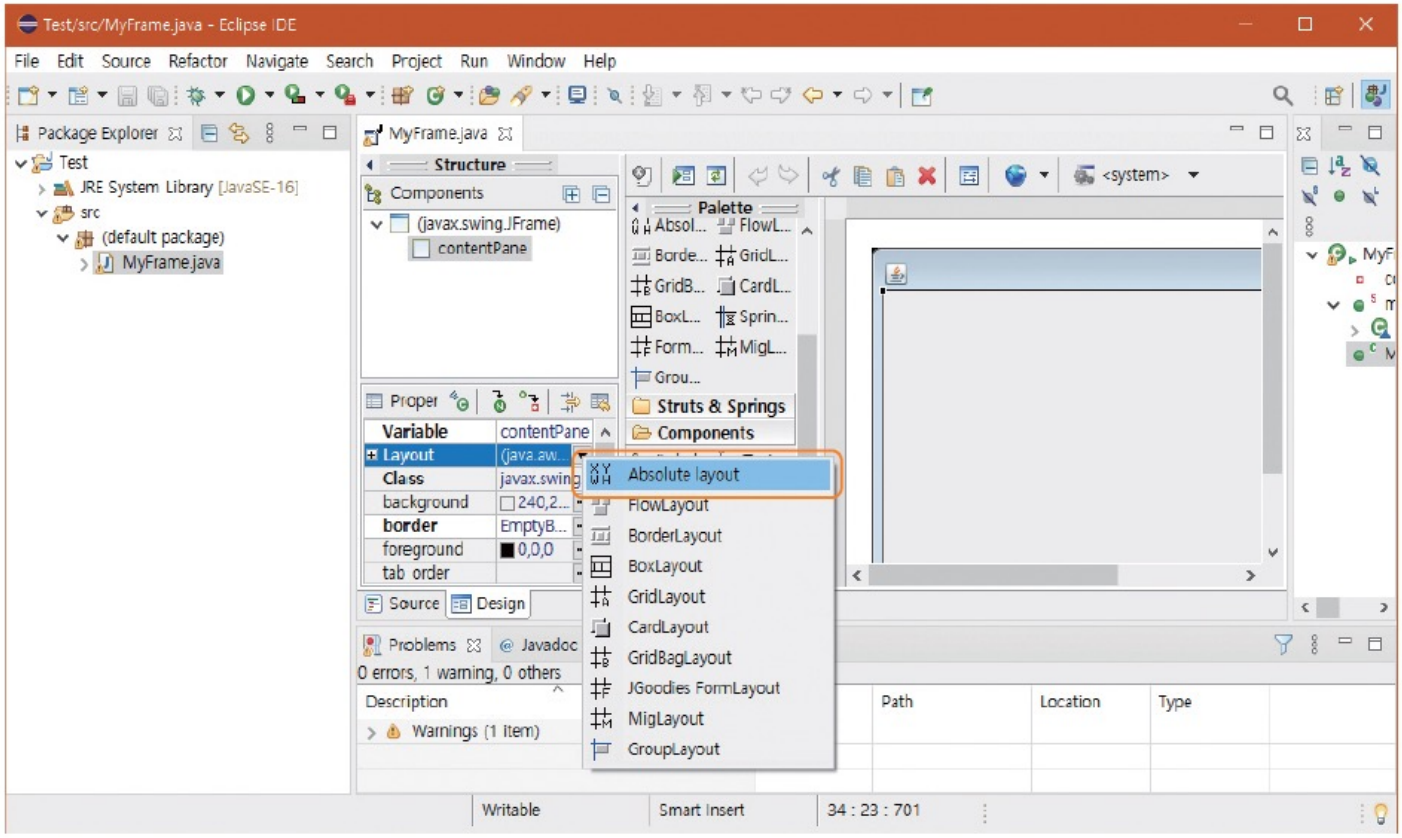
스윙 비주얼 디자이너: WindowBuilder

- 새로운 자바 프로젝트 ”Test”를 생성한다. 이어서 이클립스의 툴바에서 [JFrame]을 선택한다.



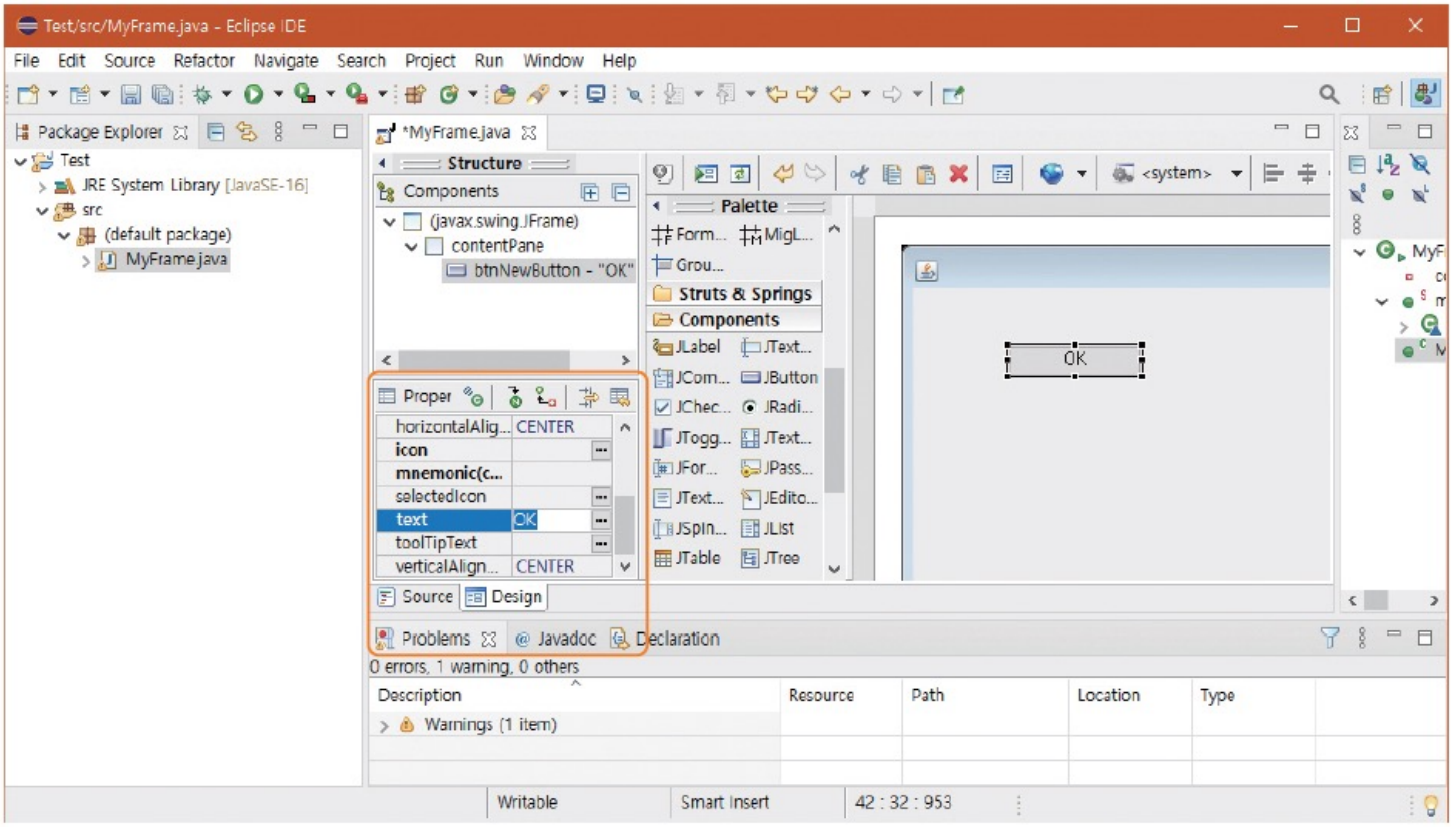
스윙 비주얼 디자이너: WindowBuilder

- [Design] 탭을 이용하여 마우스를 사용해 자유롭게 디자인 수정이 가능하다. 아무래도 코드로 화면을 디자인하는 것보다 훨씬 편리하게 화면을 설계할 수 있다.



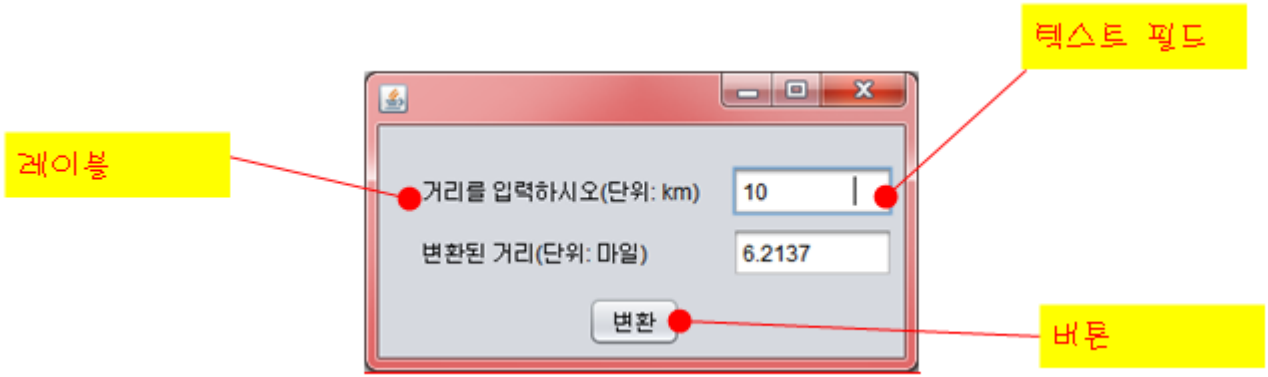
스윙 비주얼 디자이너: WindowBuilder

- 팔레트 중에서 [Components]에서 버튼을 선택하여 화면으로 이동한다. 버튼의 속성 중에서 [text] 속성을 “OK”로 변경해보자.



기초 컴포넌트

- 레이블(JLabel) - 텍스트를 표시할 수 있는 공간
- 텍스트필드(JTextField) - 사용자가 한 줄의 텍스트를 입력할 수 있는 공간
- 버튼(JButton) - 클릭되면 어떤 동작을 실행하는 버튼



레이블

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

예제: 레이블의 폰트 변경하기

```
public class LabelTest extends JFrame {
    private JPanel panel;
    private JLabel label1, label2;

    public LabelTest() {
        setTitle("레이블 테스트");
        setSize(400,150);

        panel = new JPanel();
        label1 = new JLabel("Color Label");
        label1.setForeground(Color.BLUE);
        label2 = new JLabel("Font Label");
        label2.setFont(new Font("Arial", Font.ITALIC, 30));
        label2.setForeground(Color.ORANGE);
        panel.add(label1);
        panel.add(label2);
        add(panel);
        setVisible(true);
    }

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

레이블에 이미지 표시하기

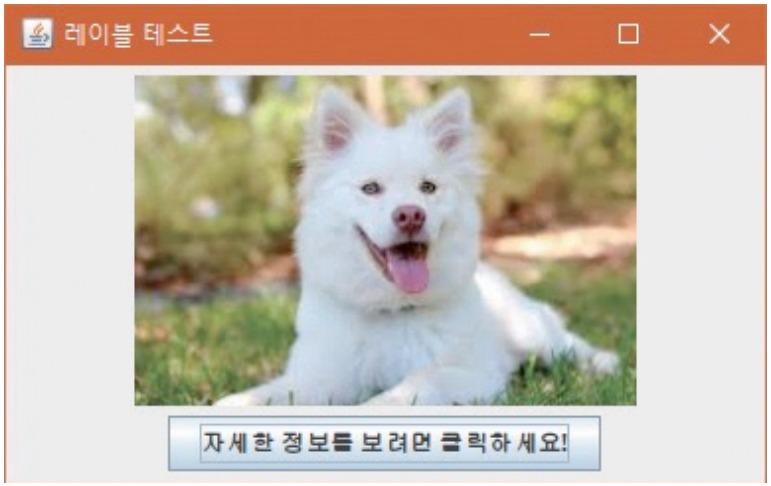
- 레이블과 버튼에는 텍스트뿐만 아니라 이미지도 표시할 수 있다.

그림 9.10 이미지를 가지고 있는 레이블과 버튼

```
ImageIcon image = new ImageIcon("d://dog.png");
JLabel label = new JLabel("Dog");
label.setIcon(image);
```

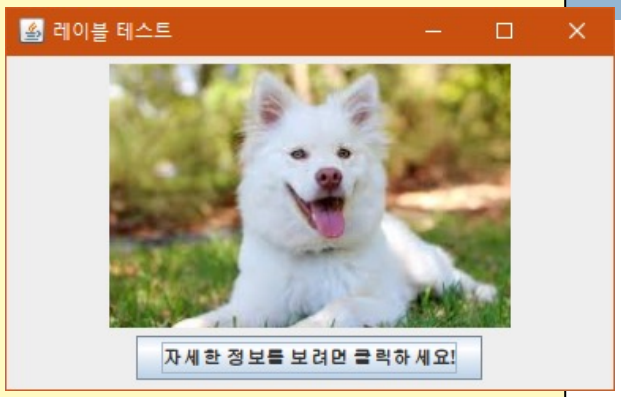
예제:

- 레이블로 이미지를 표시하고 아래에 버튼을 표시해보자. d: 드라이브에 dog.png 파일이 있어야 한다.



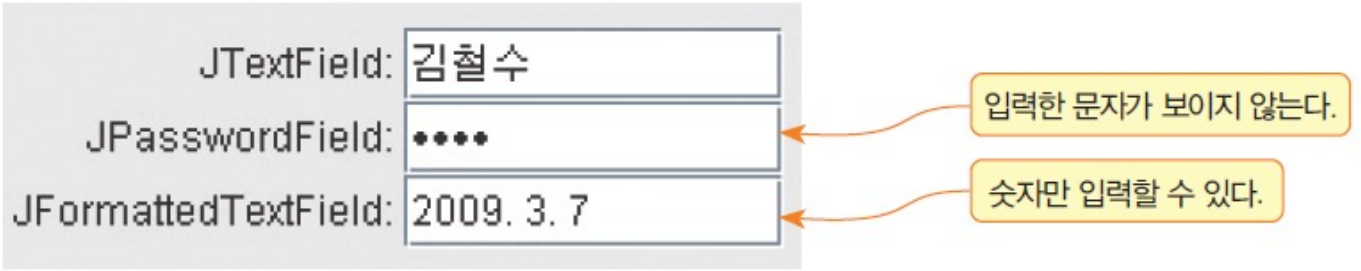
예제:

```
public class ImageLabelTest extends JFrame {
        private JPanel panel;
        private JLabel label;
        private JButton button;

        public ImageLabelTest() {
                setTitle("레이블 테스트");
                setSize(400, 250);

                panel = new JPanel();
                label = new JLabel("Dog");
                ImageIcon icon = new ImageIcon("d://dog.png");
                label.setIcon(icon);

                button = new JButton("자세한 정보를 보려면 클릭하세요!");
                panel.add(label);
                panel.add(button);
                add(panel);
                setVisible(true);
        }
        public static void main(String[] args) {
                ImageLabelTest t = new ImageLabelTest();
        }
}
```



텍스트 필드

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

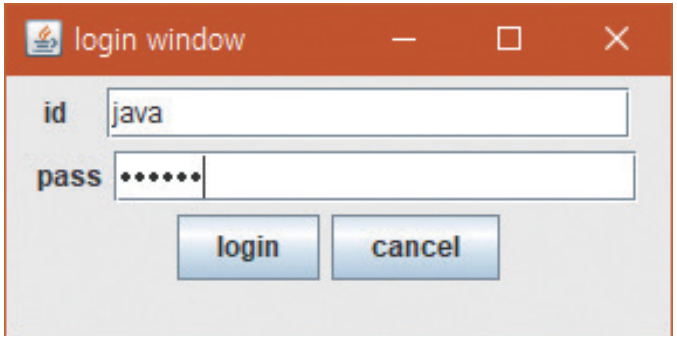



```
JTextField tf = new JTextField(30);      // 30자 크기의 텍스트 필드를 만든다.
tf.setText("아이디를 입력하시오.");          // 텍스트 필드의 텍스트를 설정한다.
System.out.println(tf.getText());        // 텍스트 필드의 텍스트를 가져온다.
```

```
tf.requestFocus();
```

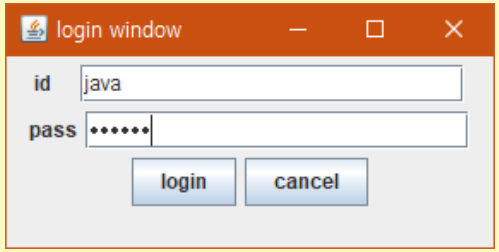
예제: 패스워드 필드 사용하기

- 패스워드 필드는 용어 그대로 암호를 입력받을 때 사용한다. 패스워드 필드에 사용자가 암호를 입력하면 글자들이 모두 * 문자로 표시된다. 패스워드 필드를 사용하여서 다음과 같은 로그인 윈도우를 작성하여 보자.



예제: 패스워드 필드 사용하기

```
public class LoginWindow extends JFrame {

        public LoginWindow()
        {
                setTitle("login window");
                setSize(300, 150);

                JPanel panel = new JPanel();
                add(panel);

                panel.add(new JLabel("id    "));
                panel.add(new JTextField(20));
                panel.add(new JLabel("pass"));
                panel.add(new JPasswordField(20));

                JButton login = new JButton("login");
                panel.add(login);

                JButton cancel = new JButton("cancel");
                panel.add(cancel);

                setVisible(true);
        }
```

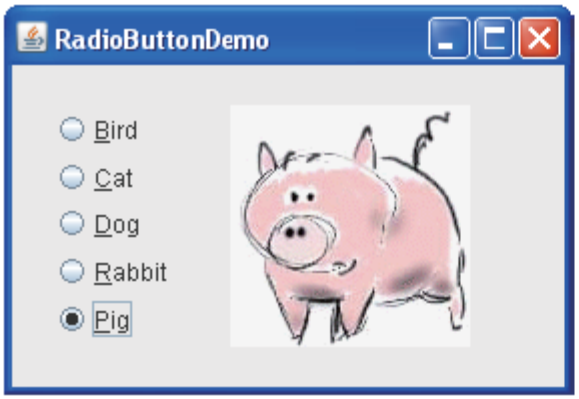


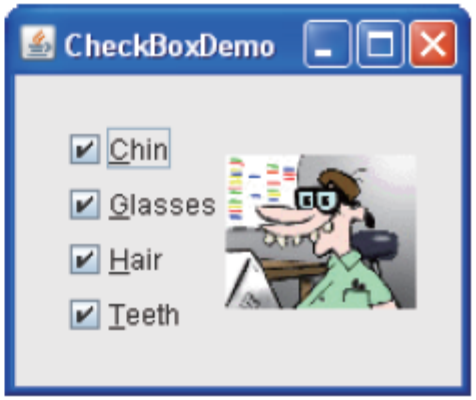
버튼

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

버튼의 종류

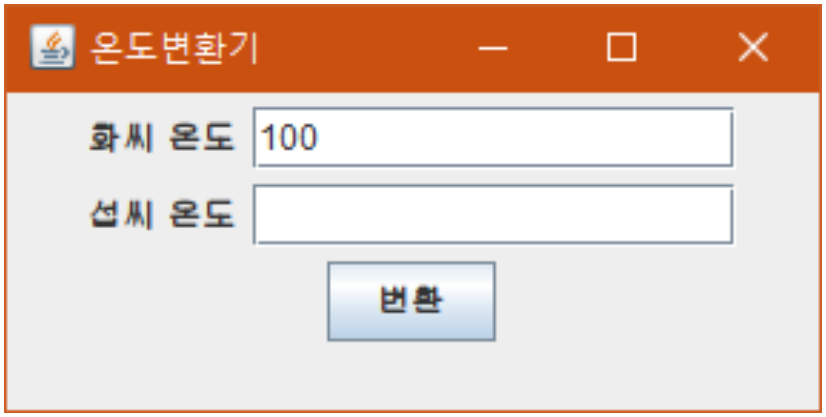
- JButton – 가장 일반적인 버튼이다.
- JCheckBox – 체크박스 버튼
- JRadioButton – 라디오 버튼으로 그룹 중의 하나의 버튼만 체크할 수 있다.
- JToggleButton – 2가지 상태를 가지고 토글이 가능한 버튼이다.





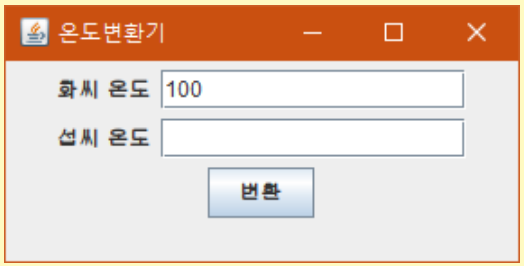
예제: 온도 변환기

- 이제까지 학습한 내용을 바탕으로 화씨 온도를 섭씨 온도로 변환해주는 애플리케이션을 작성하여 보자.



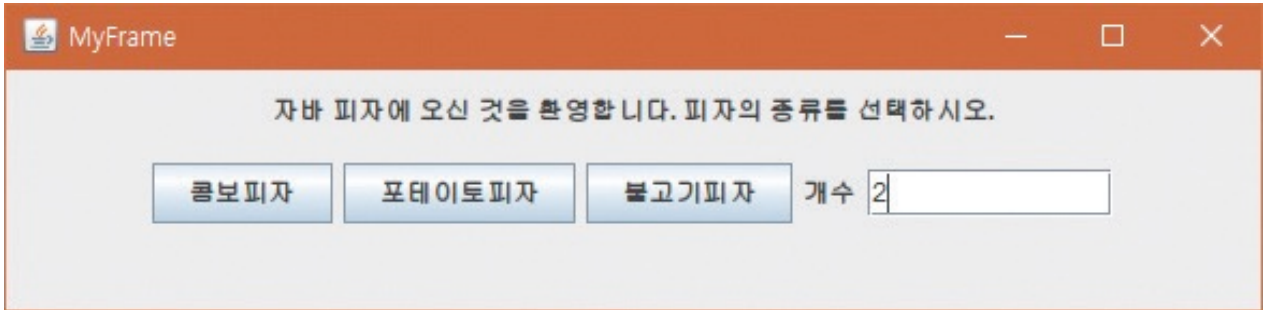
예제: 온도 변환기

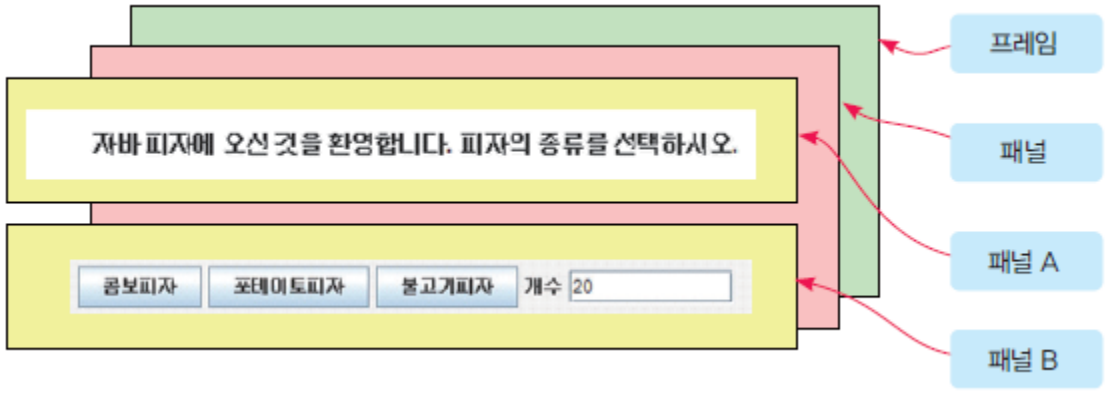
```
public class TempConverter extends JFrame {
        public TempConverter() {
                JPanel panel = new JPanel();                    // (1)
                add(panel);

                JLabel label1 = new JLabel("화씨 온도");                    // (2)
                JLabel label2 = new JLabel("섭씨 온도");
                JTextField field1 = new JTextField(15);
                JTextField field2 = new JTextField(15);
                JButton button = new JButton("변환");

                panel.add(label1);
                panel.add(field1);
                panel.add(label2);
                panel.add(field2);
                panel.add(button);

                setSize(300, 150);                              // (3)
                setTitle("온도변환기");
                setVisible(true);
        }
        public static void main(String argv[]) {
                TempConverter f = new TempConverter();
        }
}
```



예제: 피자 주문 화면 만들기

- 패널 안에 다른 패널이 포함될 수 있다. 이것을 이용하여서 다음 그림처럼 프로그램의 화면을 디자인하라.





예제: 피자 주문 화면 만들기

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

public class MyFrame extends JFrame {

        public MyFrame() {
                setSize(600, 150);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setTitle("MyFrame");

                JPanel panel = new JPanel();
                JPanel panelA = new JPanel();
                JPanel panelB = new JPanel();

                JLabel label1 = new JLabel("자바 피자에 오신 것을 환영합니다. 피자의
종류를 선택하시오.");
                panelA.add(label1);

                JButton button1 = new JButton("콤보피자");
                JButton button2 = new JButton("포테이토피자");
                JButton button3 = new JButton("불고기피자");
                panelB.add(button1);
                panelB.add(button2);
                panelB.add(button3);
```

예제: 피자 주문 화면 만들기

```
                JLabel label2 = new JLabel("개수");
                JTextField field1 = new JTextField(10);
                panelB.add(label2);
                panelB.add(field1);

                panel.add(panelA);
                panel.add(panelB);
                add(panel);
                setVisible(true);
        }

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

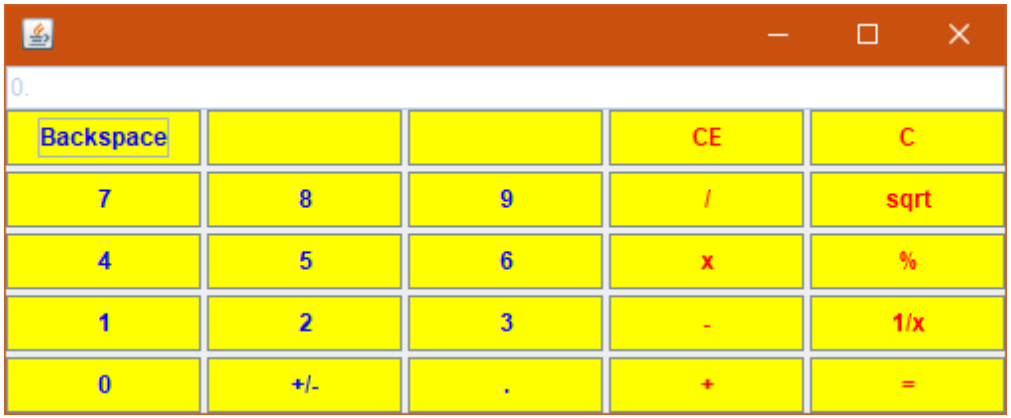
중간점검

중간점검

1. 패스워드를 받을 때 사용하는 텍스트 필드 클래스는 무엇인가?
2. 레이블에 이미지를 표시하는 절차를 설명하라.
3. 텍스트 필드에서 사용자가 입력한 텍스트를 가져오는 메소드는 무엇인가?

Lab: 계산기 예제

- 간단한 계산기를 작성하여 보자. 계산 기능은 나중에 추가하기로 하자. 여기서는 다음과 같은 화면만 구현하면 된다. 배치 관리자로 GridLayout을 사용하여 보자.



Lab: 계산기 예제

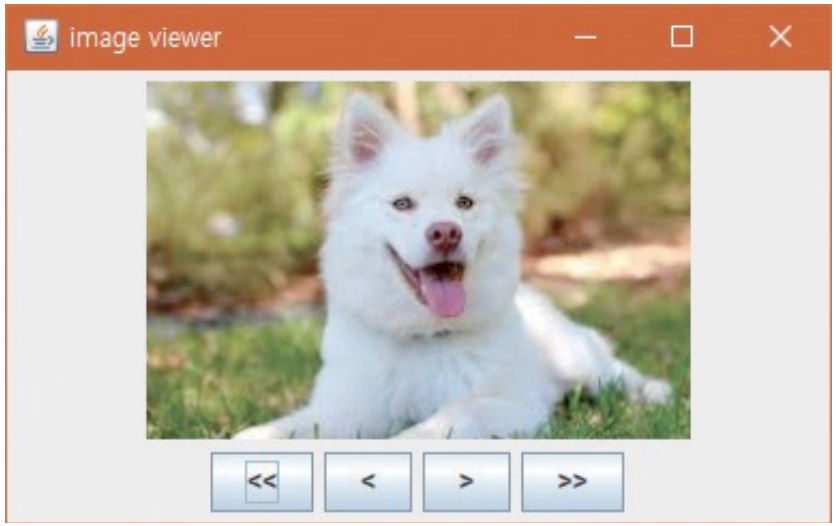
```
public class Calculator extends JFrame {

        private JPanel panel;
        private JTextField tField;
        private JButton[] buttons;
        private String[] labels = {
                        "Backspace", "", "", "CE", "C",
                        "7", "8", "9", "/", "sqrt",
                        "4", "5", "6", "x", "%",
                        "1", "2", "3", "-", "1/x",
                        "0", "+/-", ".", "+", "=",
        };


        public Calculator() {
                tField = new JTextField(35);
                panel = new JPanel();
                tField.setText("0.");
                tField.setEnabled(false);

                panel.setLayout(new GridLayout(0, 5, 3, 3));
                buttons = new JButton[25];
                int index = 0;
```

Lab: 계산기 예제

```
                    for (int rows = 0; rows < 5; rows++) {
                        for (int cols = 0; cols < 5; cols++) {
                            buttons[index] = new JButton(labels[index]);
                            if( cols >= 3 )
                              buttons[index].setForeground(Color.red);
                            else
                              buttons[index].setForeground(Color.blue);
                            buttons[index].setBackground(Color.yellow);
                            panel.add(buttons[index]);
                            index++;
                        }
                    }
                    add(tField, BorderLayout.NORTH);
                    add(panel, BorderLayout.CENTER);
                    setVisible(true);
                    pack();
            }

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

Mini Project: 사진 뷰어 만들기

- 우리는 이번 장에서 각 컴포넌트들을 화면에 배치하는 방법과 레이블에 이미지를 표시하는 방법을 학습하였다. 이것을 이용하여서 다음과 같은 이미지 뷰어를 만들어보자. 물론 아직은 이벤트 처리가 되지 않지만, 다음 장을 미리 예습한 독자라면 이벤트 처리 기능도 추가할 수 있을 것이다



Summary

- 자바에서 GUI를 위한 패키지에는 AWT, 스윙(SWING) 등이 있다. 스윙은 경량 컴포넌트들로 구성된 GUI로서 운영체제가 다르더라도 동일한 모양을 지원한다.
- 컨테이너는 안에 컴포넌트들을 저장할 수 있는 특수한 컴포넌트이다.
- JFrame은 최상위 컨테이너로서 윈도우 기능을 제공한다. 내부에 있는 컨텐트 페인을 이용하여서 컴포넌트들을 부착할 수 있다.
- 컨테이너는 하나의 배치 관리자를 가질 수 있다. 배치 관리자는 컴포넌트들의 위치와 크기를제어한다.
- FlowLayout은 컨테이너에 추가되는 순서대로 컴포넌트를 부착한다. 위쪽에서 아래쪽으로, 왼쪽에서 오른쪽으로 배치한다. 패널의 기본 배치 관리자이다.
- BorderLayout은 컨테이너의 영역을 동서남북, 중앙의 5개의 영역으로 구분하여 이 영역에 컴포넌트를 배치한다. 프레임의 기본 배치 관리자이다.
- 스윙 컴포넌트에 이미지를 표시하려면 먼저 ImageIcon 인스턴스를 생성하여야 한다. 이후에 setIcon() 메소드를 사용하여 컴포넌트에 이미지를 지정한다.
- 텍스트 필드 중에서 JPasswordField를 사용하면 사용자가 입력하는 텍스트가 보이지 않는다.

Q & A