

{ JAVA }

선린인터넷  
웹 운영 과

# 그래픽 사용자 인터페이스 : GUI-2

- AWT 및 Swing
- JFrame컨테이너
- 배치관리자

# GUI : AWT Swing

## 자바의 GUI (Graphical User Interface)

### ❖ GUI 장점 vs CUI

- 사용자에게 이해하기 쉽도록 그래픽을 이용하여 정보 제공
- 사용자는 마우스, 키보드 등을 이용하여 쉽게 입력

### ❖ 자바 GUI 프로그래밍

- AWT와 Swing 패키지에서 제공하는 GUI 컴포넌트, 그래픽 등 메커니즘 이용
  - AWT – java.awt 패키지
  - Swing – javax.swing 패키지

## AWT 와 Swing

### ❖ 자바에서의 GUI 화면 구현 방법

#### ■ AWT : Abstract Window Toolkit 이용

- java.awt 패키지 클래스 이용
- 운영체제의 자원을 이용하여 화면을 구현
- 각 운영체제에 종속적 → 운영체제의 GUI 컴포넌트의 도움을 받아 작동

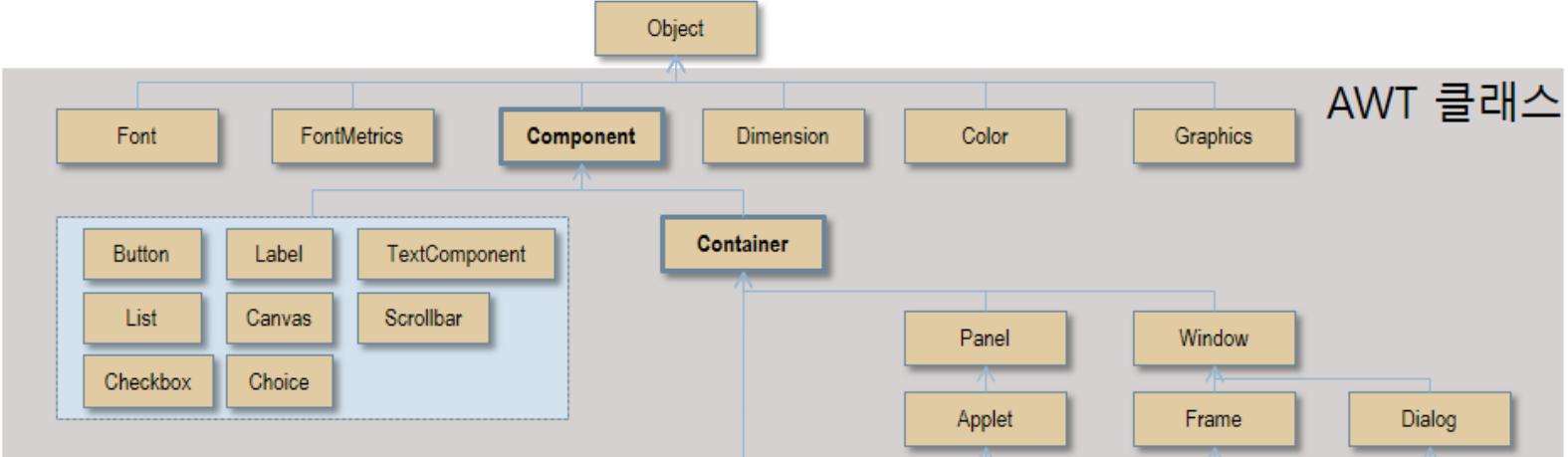
#### ■ Swing 이용

- javax.swing 패키지 클래스 이용
- 자체 클래스의 기능으로 화면을 구현
- 각 운영체제에 독립적 → 운영체제에 의존하지 않음 : Swing 컴포넌트 = 경량 컴포넌트
- 클래스 이름이 "J"로 시작하면, 스윙 클래스를 의미함

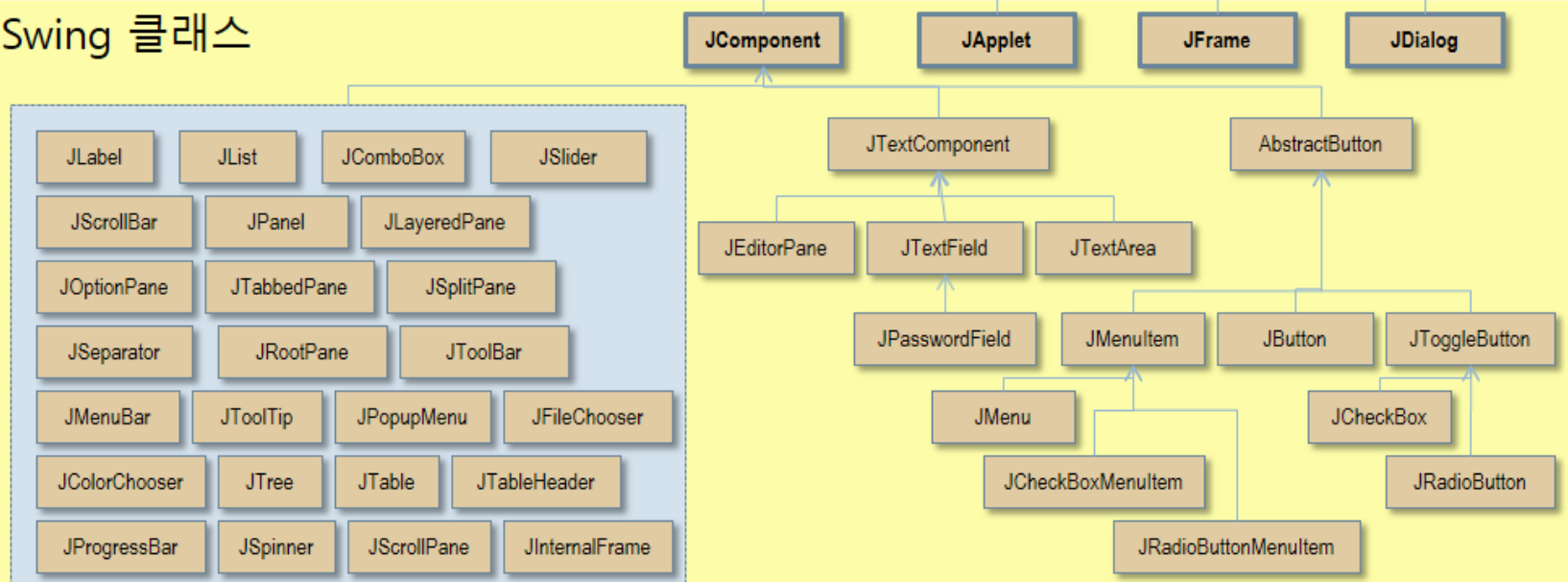
# GUI

## GUI API

### 계층 구조



### Swing 클래스



## Swing컴포넌트



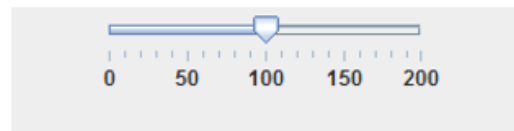
JButton



JCheckBox



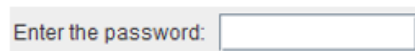
JRadioButton



JSlider



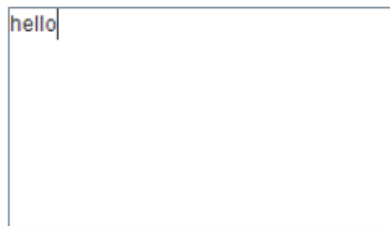
JTextField



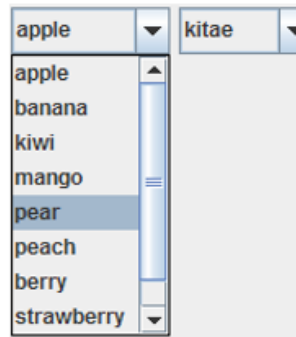
JPasswordField



JSpinner



JTextArea

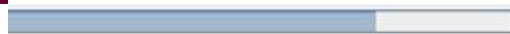


JComboBox

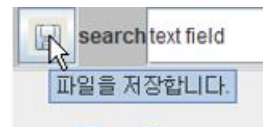


JList

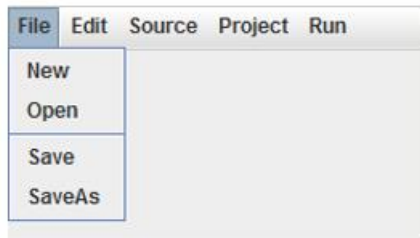
## Swing컴포넌트



JProgressBar



JToolTip



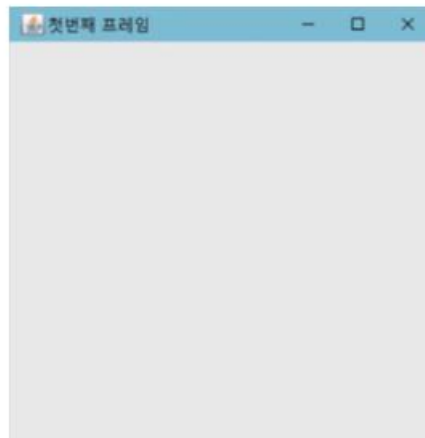
JMenu



JScrollPane



JDialog







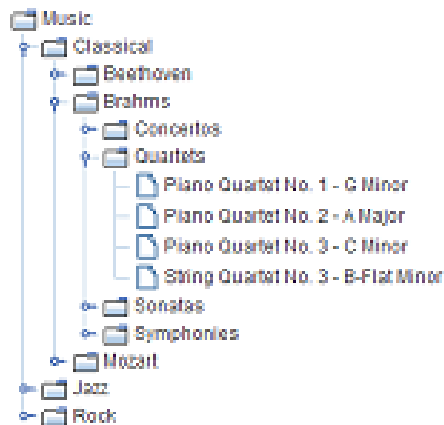
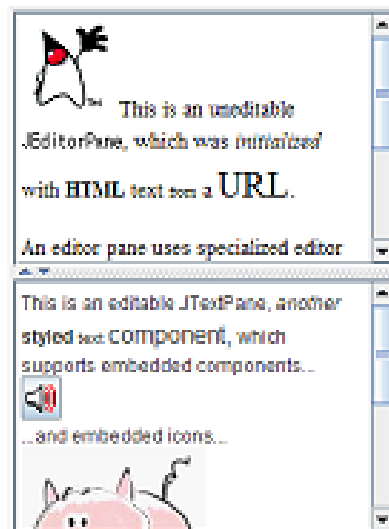
JFrame



JColorChooser

## Swing컴포넌트

First Name	Last Name	Favorite Color	Favorite Movie	Favorite Number	Favorite Food
Nike	Albers	Green	Brazil	44	
Mark	Andrews	Blue	Curse of the Dam...	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	

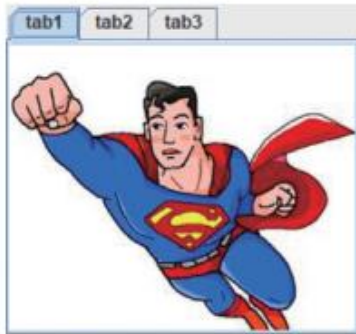
*JTable**JTree**JEditorPane and JTextPane*



# Swing컴포넌트



*JToolBar*



*JTabbedPane*

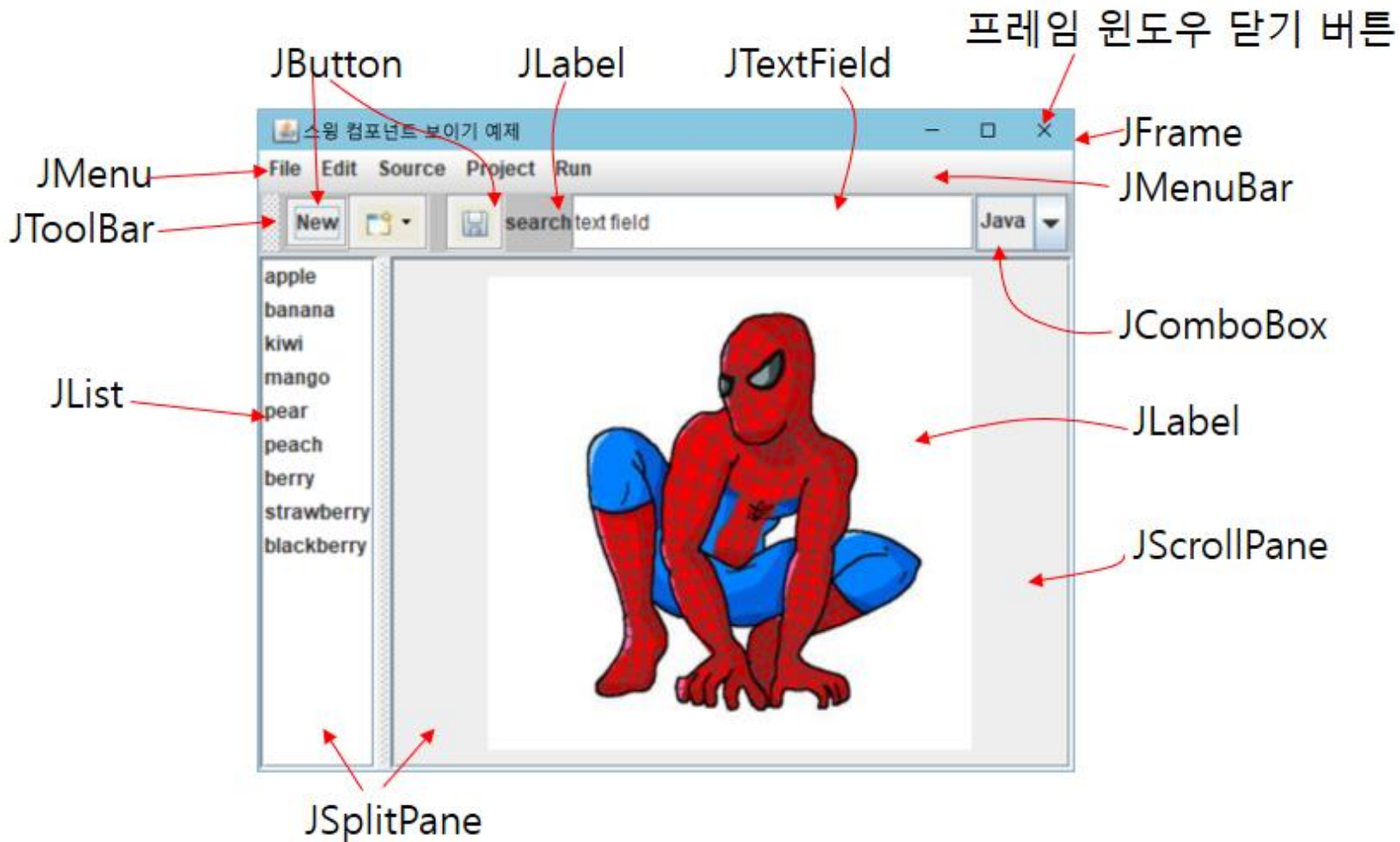


*JSplitPane*

# GUI : 마치 레고 블록처럼 스윙 컴포넌트를 조립해 작성

## GUI 샘플

### Composite 디자인패턴



# 컴포넌트 컨테이너

## Component 와 Container

### ❖ 자바 GUI 응용프로그램 : 컴포넌트들(GUI 객체들)로 구성됨

- 다른 컴포넌트를 포함할 수 있는 지 여부 → 단순 컴포넌트, 컨테이너로 분류됨
- 빈 판 위에 레고 블록을 쌓아 가듯이, GUI 컴포넌트들을 쌓아 GUI 프로그램 구성

### ❖ 단순 컴포넌트

- 컨테이너와 달리 다른 컴포넌트를 포함할 수 없음
- 컨테이너에 포함되어야 비로소 화면에 출력될 수 있음
- JButton, JLabel, JCheckBox, JList, JMenu, JTextField, JScrollbar, JTextArea 등

### ❖ 컨테이너

- 다른 GUI 컴포넌트를 포함할 수 있는 컴포넌트 : JFrame, JDialog, JApplet 등

## Component 와 Container

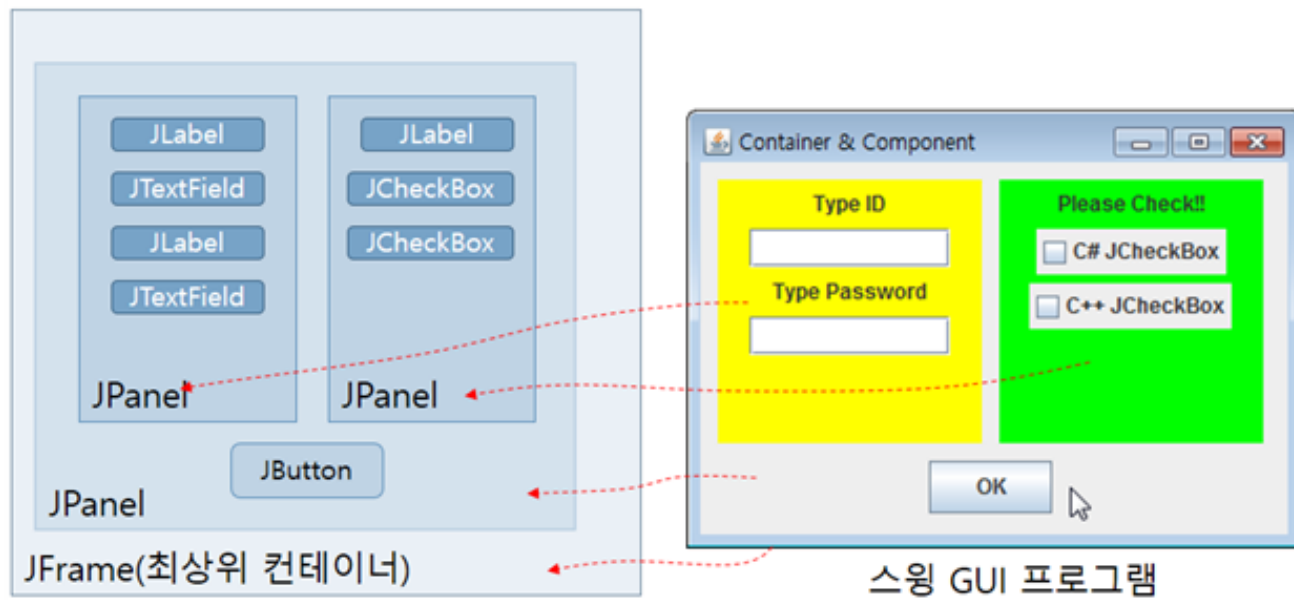
❖ 컨테이너 : 최상위 컨테이너와 일반적인 컨테이너로 나뉨

- 최상위 컨테이너 : 다른 컨테이너에 포함될 수 없음
  - JFrame, JDialog, JApplet
- 일반적인 컨테이너 : 다른 컨테이너에 포함될 수 있음
  - JPanel, JScrollPane 등

컨테이너는 컴포넌트들을 넣을 수 있는 상자 같은 것...

## Component 와 Container

### 컨테이너와 컴포넌트의 포함관계



# JFrame 컨테이너

## JFrame

### ❖ 모든 스윙 컴포넌트를 담는 최상위 GUI 컨테이너

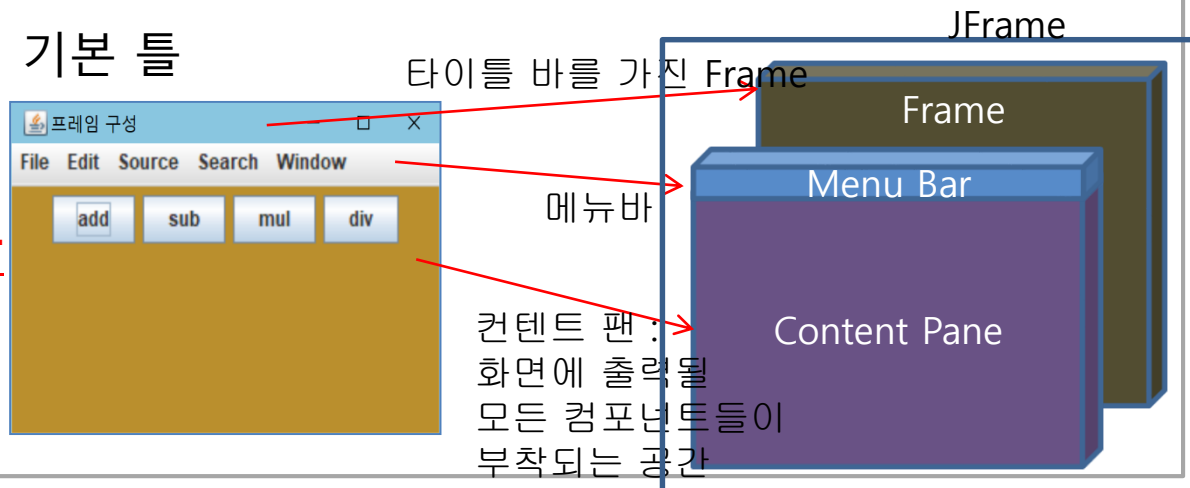
- 컴포넌트가 화면에 보이려면 프레임에 부착되어야 함
- 프레임을 닫으면 프레임 내 모든 컴포넌트가 보이지 않게 됨

### ❖ JFrame기본구성

- 프레임 - 스윙 프로그램의 기본 틀
- 메뉴바 - 메뉴 부착



- **컨텐츠팬 - 컴포넌트 부착**



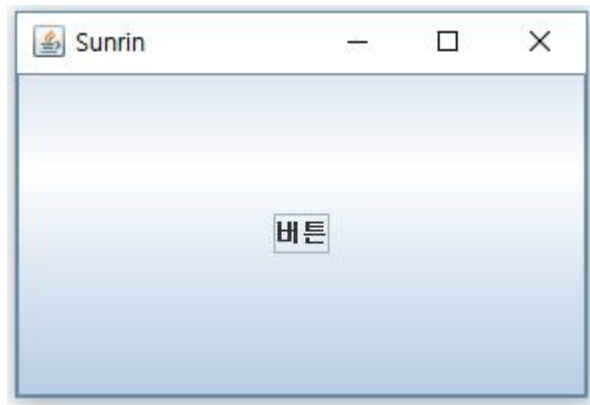


1번째

윈도우창을 만듭시다  
버튼을 부착해봅시다

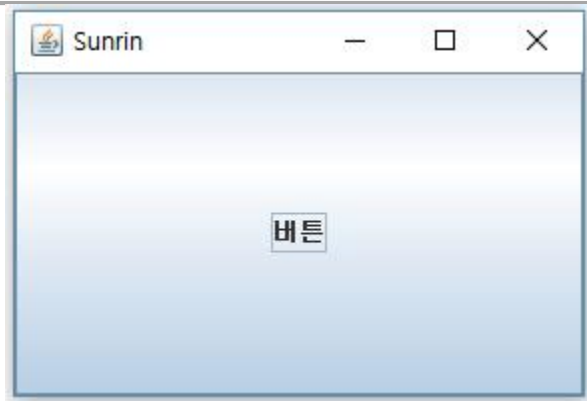
# GUI 맛보기

## Exam1 : 버튼 1개가 부착된 윈도우창 1개 만들기



## Exam1 : 버튼 1개가 부착된 윈도우창 1개 만들기

```
import java.awt.Container;
import javax.swing.*;
public class Exam1 extends JFrame {
    public Exam1() {
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Sunrin");
        setLocation(200, 200); // 프레임의 위치를 정함
        // 컨텐트팬을 가져와서, 버튼을 배치함
        Container contentPane = getContentPane();
        JButton button = new JButton("버튼");
        contentPane.add(button); // 컨텐트팬에 배치함
        // 자바5부터 컨텐트팬을 거쳐 배치하는 작업을 생략하고
        // frame에 바로 배치 가능함 (단, 컨텐트팬에 없어진 게 아님)
        setVisible(true);
    }
    public static void main(String[] args) {
        new Exam1();
    }
}
```

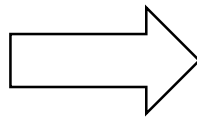
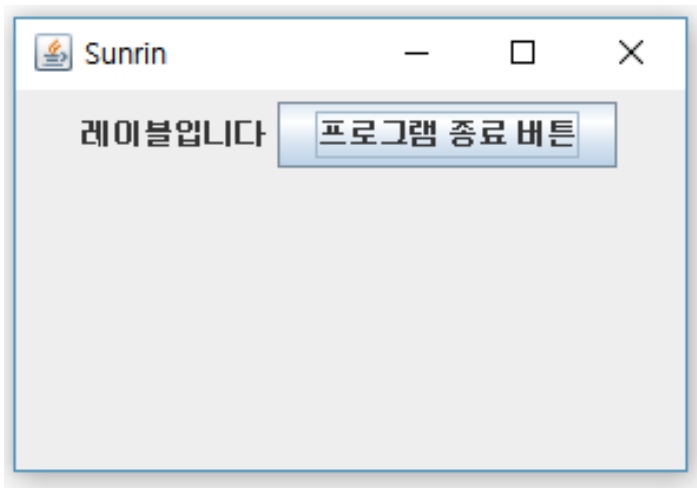


2번째

레이블 1개 추가, 레이아웃(FlowLayout으로) 바꾸기 그리고...

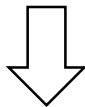
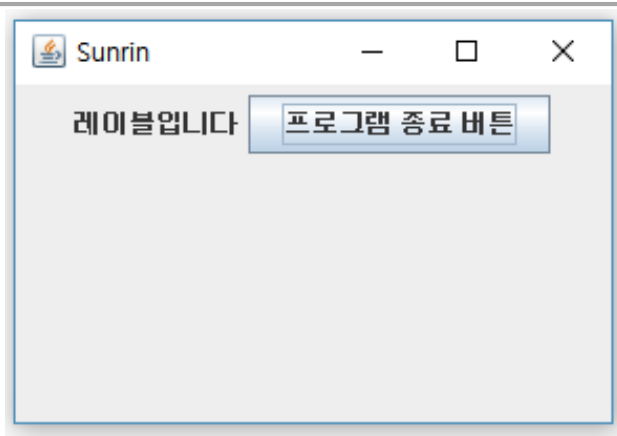
버튼 클릭시 프로그램  
종료하도록 해봅시다

**Exam1 : 레이블 1개 추가, 버튼 클릭시 프로그램 종료 → `System.exit(0);` 활용하자!**



Exam1 : 레이블 1개 추가, 버튼 클릭시 프로그램 종료 → `System.exit(0);` 활용하자!

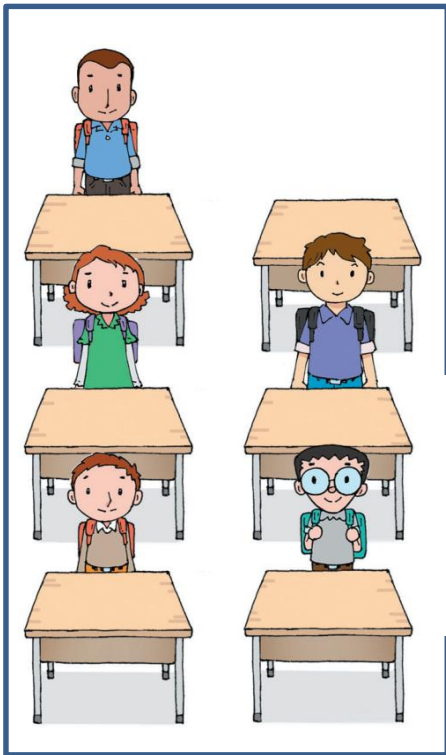
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Exam2 extends JFrame implements ActionListener {
    public Exam2() {
        setSize(300, 200); // 프레임의 사이즈(가로,세로)를 정함
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Sunrin");// 프레임의 타이틀을 "Sunrin"으로 함
        JLabel label = new JLabel("레이블입니다");
        this.add(label); // 레이블 객체를 프레임에 배치함
        JButton button = new JButton("프로그램 종료 버튼");
        add(button); // 버튼 객체를 프레임에 배치함
        button.addActionListener(this); //버튼에 액션리스너 객체 등록
        setLayout(new FlowLayout()); // 배치관리자 셋팅
        setVisible(true); // 프레임을 화면에 나타나게 함
    }
    public void actionPerformed(ActionEvent e) {
        System.exit(0); // 프로그램을 종료하게 함
    }
    public static void main(String[] args) {
        new Exam2();
    }
}
```



# 컨테이너와 배치 개념

## 컨테이너와 배치 개념

컨테이너(Container)



이쪽으로  
가세요.



배치관리자  
(Layout Manager)

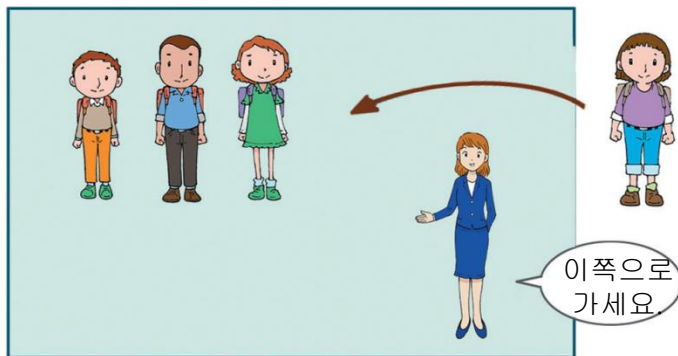
컴포넌트  
(Component)

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

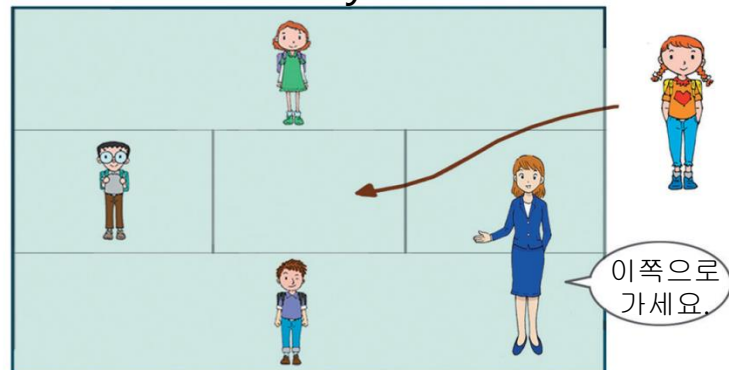


## 컨테이너와 배치 개념

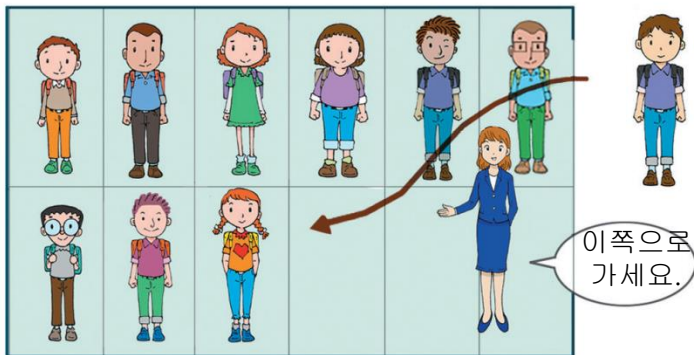
FlowLayout



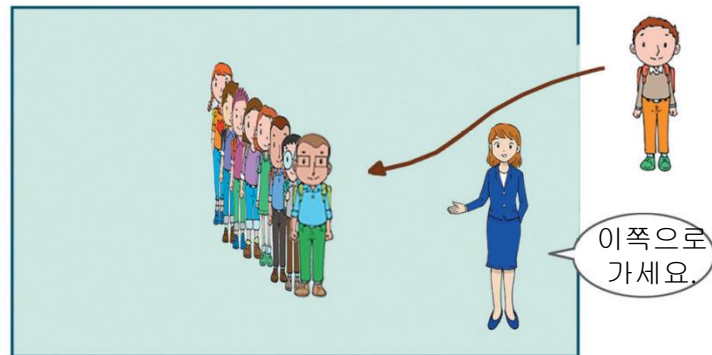
BorderLayout



GridLayout



Null : 레이아웃 없음



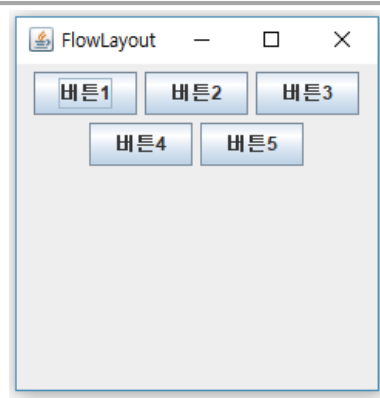
## 컨테이너와 배치 개념

※ 컨테이너와 디폴트 배치 관리자 : 교과서 200쪽

Swing 컨테이너	default Layout Manager
<u>JFrame</u>	<u>BorderLayout</u>
<u>JDialog</u>	<u>BorderLayout</u>
<u>JWindow</u>	<u>BorderLayout</u>
<u>JPanel</u>	<u>FlowLayout</u>
<u>JApplet</u>	<u>FlowLayout</u>

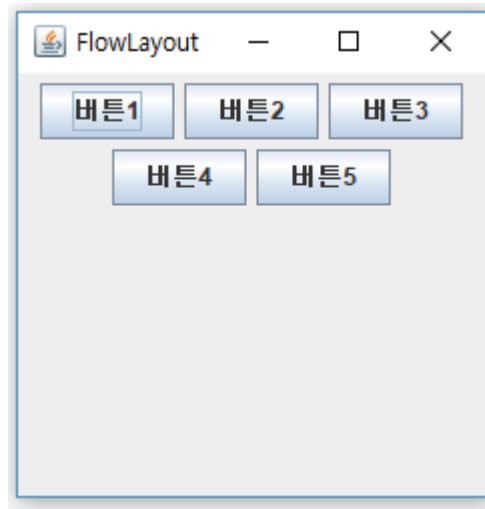
# FlowLayout

1. 왼쪽에서 오른쪽으로 배치
2. 중앙 정렬
3. 한 줄에 모든 컴포넌트 배치가 불가능하면  
다음 줄에 이어서 배치함.



## Exam3

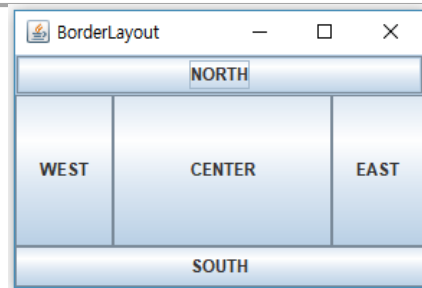
```
import java.awt.*;
import javax.swing.*;
public class Exam3 extends JFrame {
    public Exam3() {
        setTitle("FlowLayout");
        setSize(250, 250);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        // 콘텐츠 페인에 FlowLayout 배치관리자 적용하기
        c.setLayout(new FlowLayout());
        c.add(new JButton("버튼1"));
        c.add(new JButton("버튼2"));
        c.add(new JButton("버튼3"));
        c.add(new JButton("버튼4"));
        c.add(new JButton("버튼5")); //add(new JButton("버튼5")); 가능
        setVisible(true);
    }
    public static void main(String[] args) {
        new Exam3();
    }
}
```



# BorderLayout

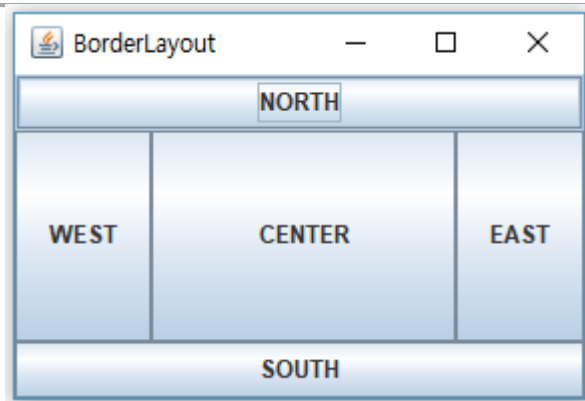
GUI창을 5개 영역으로 나누어 배치

- BorderLayout.NORTH
- BorderLayout.WEST
- BorderLayout.CENTER
- BorderLayout.EAST
- BorderLayout.SOUTH



## Exam4

```
import java.awt.*;
import javax.swing.*;
public class Exam4 extends JFrame {
    public Exam4() {
        setTitle("BorderLayout");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new BorderLayout());
        c.add(new JButton("NORTH"), BorderLayout.NORTH);
        c.add(new JButton("SOUTH"), BorderLayout.SOUTH);
        c.add(new JButton("CENTER"), BorderLayout.CENTER);
        c.add(new JButton("WEST"), BorderLayout.WEST);
        c.add(new JButton("EAST"), BorderLayout.EAST);
        setVisible(true);
    }
    public static void main(String[] args) {
        new Exam4();
    }
}
```



# GridLayout



1. 레이아웃을 생성할 때 행과 열의 개수를 지정
2. 추가된 순서대로 컴포넌트들을 1행 1열부터 배치

## Exam5

```
import java.awt.*;
public class Exam5 extends JFrame {
    public Exam5() {
        this.setTitle("GridLayout");
        this.setSize(300, 300);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container pane=this.getContentPane();
        pane.setLayout(new GridLayout(3,2));
        pane.add(new JButton("One"));
        pane.add(new JButton("Two"));
        pane.add(new JButton("Three"));
        pane.add(new JButton("Four"));
        pane.add(new JButton("Five"));
        pane.add(new JButton("Six"));
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exam5();
    }
}
```





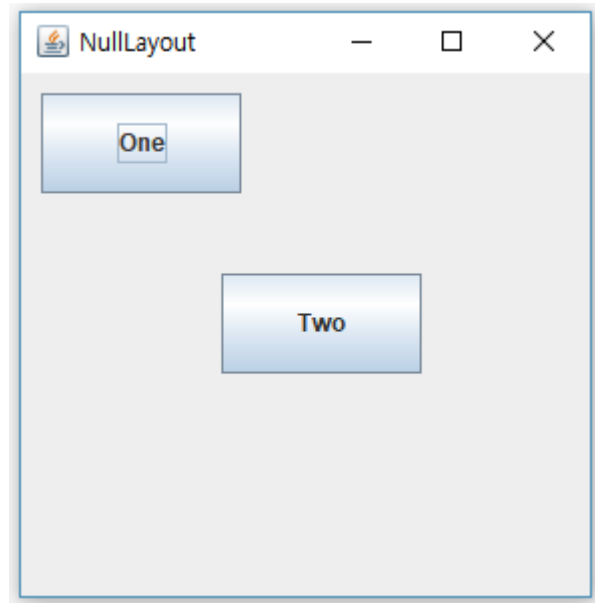
# null

## 배치관리자가 없는 컨테이너

- Layout이 없기 때문에 컴포넌트의 위치와 크기가 없으면 제대로 보이지 않음.
- 컴포넌트의 크기나 위치를 개발자 임의로 결정
- 게임 프로그램과 같이 시간이나 마우스/키보드의 입력에 따라 컴포넌트들의 위치와 크기가 수시로 변할 때 사용

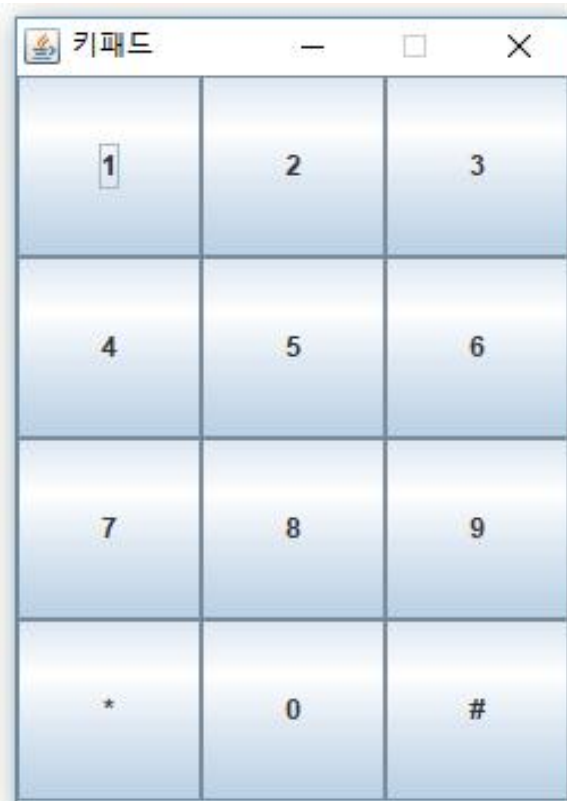
## Exam6

```
import java.awt.*;
import javax.swing.*;
public class Exam6 extends JFrame {
    public Exam6() {
        this.setTitle("NullLayout");
        this.setSize(300, 300);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container pane=this.getContentPane();
        pane.setLayout(null);
        JButton btn1=new JButton("One");
        JButton btn2=new JButton("Two");
        btn1.setBounds(10, 10, 100, 50);
        btn2.setBounds(100, 100, 100, 50);
        pane.add(btn1);
        pane.add(btn2);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exam6();
    }
}
```



# [ 실습문제 ]

## 키패드 만들기



파일명: Exam7.java

## Exam7

```
import java.awt.*;
import javax.swing.*;
public class Exam7 extends JFrame {
    public Exam7() {
        setTitle("키패드");
        setSize(250, 350);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(4, 3));
        for(int i=1;i<10;i++)
            add(new JButton(String.valueOf(i)));
        add(new JButton("*"));
        add(new JButton("0"));
        add(new JButton("#"));
        setResizable(false); // 창 크기 조절이 되지 않도록 셋팅
        setVisible(true);
    }
    public static void main(String[] args) {
        new Exam7();
    }
}
```

