# 1811943 황성미 7일차 과제

▼ **4.2.8 Panel**

컴포넌트를 포함할 수 있는 컨테이너 기능이 있음

visible한 프레임이나 창을  add시켜주어야 함

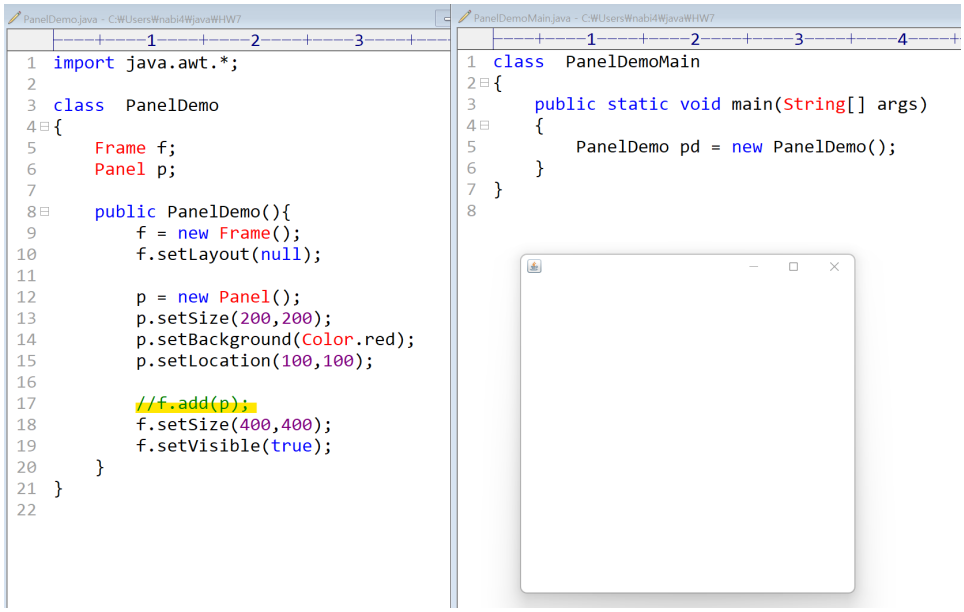
▼ Construct

Panel() , Panel(LayoutManager layout)

▼ Method

add(other components) , setSize(), setLocation()

▼ Lab1



```java
import java.awt.*;

class  PanelDemo
{
    Frame f;
    Panel p;

    public PanelDemo(){
        f = new Frame();
        f.setLayout(null);

        p = new Panel();
        p.setSize(200,200);
        p.setBackground(Color.red);
        p.setLocation(100,100);

        //f.add(p);
        f.setSize(400,400);
        f.setVisible(true);
    }
}
```

```java
class  PanelDemoMain
{
    public static void main(String[] args)
    {
        PanelDemo pd = new PanelDemo();
    }
}
```

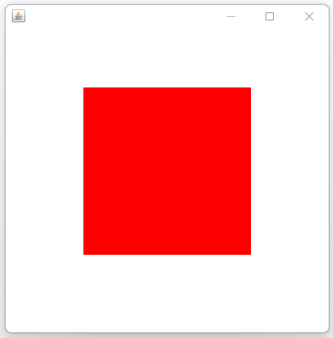
panel을 add해주지 않으면 나타나지 않음

PanelDemo.java - C:₩Users₩nabi4₩java₩HW7

```java
import java.awt.*;

class  PanelDemo
{
    Frame f;
    Panel p;

    public PanelDemo(){
        f = new Frame();
        f.setLayout(null);

        p = new Panel();
        p.setSize(200,200);
        p.setBackground(Color.red);
        p.setLocation(100,100);

        f.add(p);
        f.setSize(400,400);
        f.setVisible(true);
    }
}
```

PanelDemoMain.java - C:₩Users₩nabi4₩java₩HW7

```java
class  PanelDemoMain
{
    public static void main(String[] args)
    {
        PanelDemo pd = new PanelDemo();
    }
}
```



f.add(p); 까지 해주면 잘 나타나는 것을 볼 수 있음

Main 프로그램을 컴파일하게 되면 연관되어있는 프로그램도 자동적으로 컴파일해주기때문에 연관되어있는 프로그램을 수정해도 Main만 컴파일해주면 됨

## ▼ Login

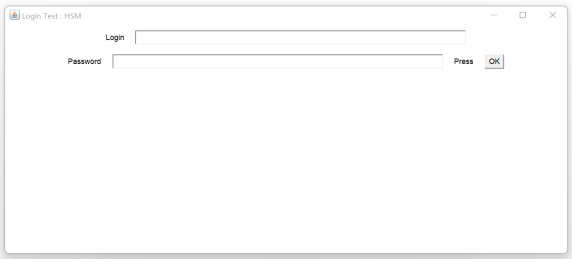
### ▼ Panel만 추가한 경우

LoginUIwithPanel.java - C:₩Users₩nabi4₩java₩HW7

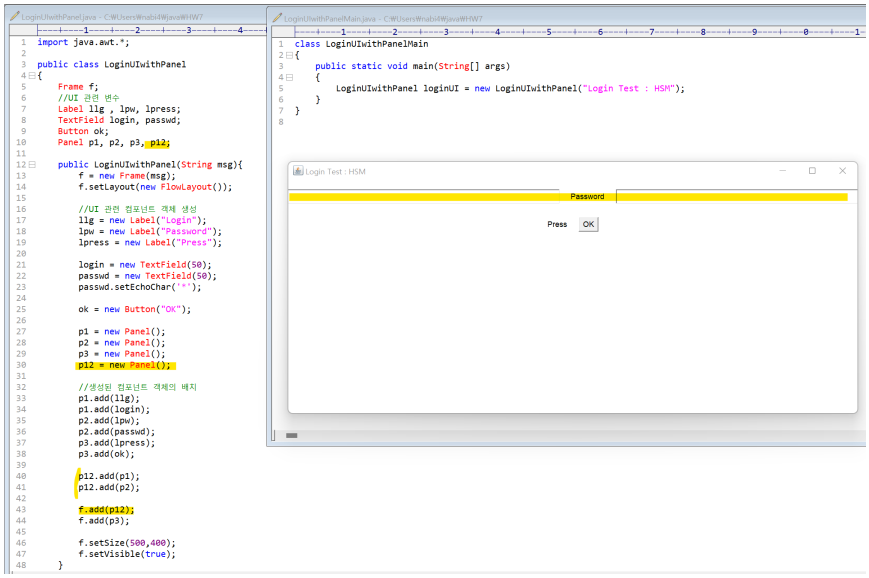
```java
import java.awt.*;

public class LoginUIwithPanel
{
    Frame f;
    //UI 관련 변수
    Label llg , lpw, lpress;
    TextField login, passwd;
    Button ok;
    Panel p1, p2, p3;

    public LoginUIwithPanel(String msg){
        f = new Frame(msg);
        f.setLayout(new FlowLayout());

        //UI 관련 컴포넌트 객체 생성
        llg = new Label("Login");
        lpw = new Label("Password");
        lpress = new Label("Press");

        login = new TextField(50);
        passwd = new TextField(50);
        passwd.setEchoChar('*');

        ok = new Button("OK");

        p1 = new Panel();
        p2 = new Panel();
        p3 = new Panel();

        //생성된 컴포넌트 객체의 배치
        p1.add(llg);
        p1.add(login);
        p2.add(lpw);
        p2.add(passwd);
        p3.add(lpress);
        p3.add(ok);

        f.add(p1);
        f.add(p2);
        f.add(p3);

        f.setSize(500,400);
        f.setVisible(true);
    }

    public LoginUIwithPanel(){
        f = new Frame("Login UI Test");
```

LoginUIwithPanelMain.java - C:₩Users₩nabi4₩java₩HW7

```java
class LoginUIwithPanelMain
{
    public static void main(String[] args)
    {
        LoginUIwithPanel loginUI = new LoginUIwithPanel("Login Test : HSM");
    }
}
```




label과 textfield , button과의 틀은 생겼지만 자유분망하게 움직이는 상태



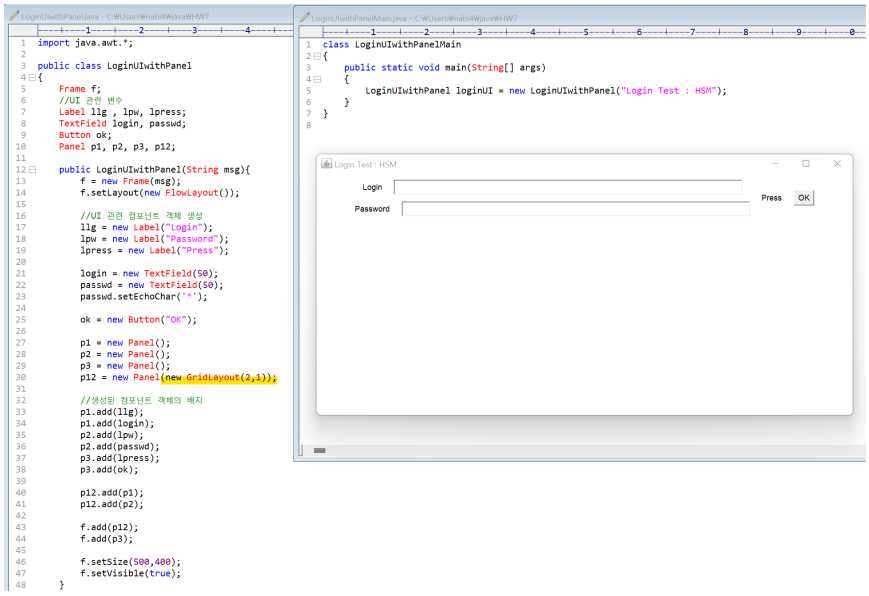
```
import java.awt.*;

public class LoginUIwithPanel
{
    Frame f;
    //UI 관련 변수
    Label llg , lpw, lpress;
    TextField login, passwd;
    Button ok;
    Panel p1, p2, p3, p12;

    public LoginUIwithPanel(String msg){
        f = new Frame(msg);
        f.setLayout(new FlowLayout());

        //UI 관련 컴포넌트 객체 생성
        llg = new Label("Login");
        lpw = new Label("Password");
        lpress = new Label("Press");

        login = new TextField(50);
        passwd = new TextField(50);
        passwd.setEchoChar('*');

        ok = new Button("OK");

        p1 = new Panel();
        p2 = new Panel();
        p3 = new Panel();
        p12 = new Panel();

        //생성된 컴포넌트 객체의 배치
        p1.add(llg);
        p1.add(login);
        p2.add(lpw);
        p2.add(passwd);
        p3.add(lpress);
        p3.add(ok);

        p12.add(p1);
        p12.add(p2);

        f.add(p12);
        f.add(p3);

        f.setSize(500,400);
        f.setVisible(true);
    }
```

```
class LoginUIwithPanelMain
{
    public static void main(String[] args)
    {
        LoginUIwithPanel loginUI = new LoginUIwithPanel("Login Test : HSM");
    }
}
```



p12을 추가하여 p1과 p2 panel을 묶어주었더니 옆으로 붙어버림

▼ Panel에 GridLayout을 준 경우



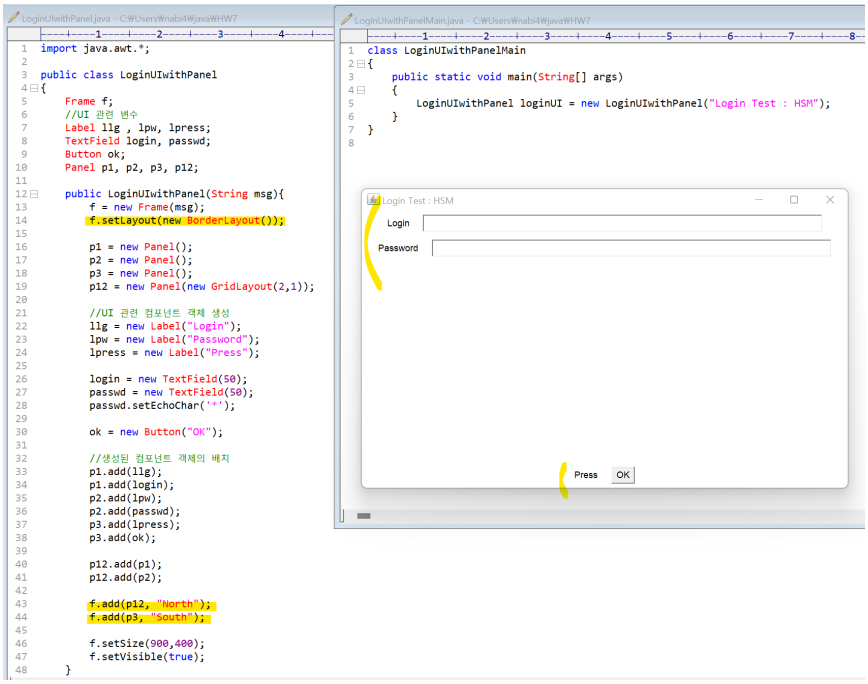
```
import java.awt.*;

public class LoginUIwithPanel
{
    Frame f;
    //UI 관련 변수
    Label llg , lpw, lpress;
    TextField login, passwd;
    Button ok;
    Panel p1, p2, p3, p12;

    public LoginUIwithPanel(String msg){
        f = new Frame(msg);
        f.setLayout(new FlowLayout());

        //UI 관련 컴포넌트 객체 생성
        llg = new Label("Login");
        lpw = new Label("Password");
        lpress = new Label("Press");

        login = new TextField(50);
        passwd = new TextField(50);
        passwd.setEchoChar('*');

        ok = new Button("OK");

        p1 = new Panel();
        p2 = new Panel();
        p3 = new Panel();
        p12 = new Panel(new GridLayout(2,1));

        //생성된 컴포넌트 객체의 배치
        p1.add(llg);
        p1.add(login);
        p2.add(lpw);
        p2.add(passwd);
        p3.add(lpress);
        p3.add(ok);

        p12.add(p1);
        p12.add(p2);

        f.add(p12);
        f.add(p3);

        f.setSize(500,400);
        f.setVisible(true);
    }
```

```
class LoginUIwithPanelMain
{
    public static void main(String[] args)
    {
        LoginUIwithPanel loginUI = new LoginUIwithPanel("Login Test : HSM");
    }
}
```



GridLayout(2,1)을 설정하여 Login 밑에 Password가 올 수 있게 되었음. 이제 p3만 조정해 주면 된다!

▼ Frame에 BorderLayout을 준 경우



```java
import java.awt.*;

public class LoginUIwithPanel
{
    Frame f;
    //UI 관련 변수
    Label llg , lpw, lpress;
    TextField login, passwd;
    Button ok;
    Panel p1, p2, p3, p12;

    public LoginUIwithPanel(String msg){
        f = new Frame(msg);
        f.setLayout(new BorderLayout());

        p1 = new Panel();
        p2 = new Panel();
        p3 = new Panel();
        p12 = new Panel(new GridLayout(2,1));

        //UI 관련 컴포넌트 객체 생성
        llg = new Label("Login");
        lpw = new Label("Password");
        lpress = new Label("Press");

        login = new TextField(50);
        passwd = new TextField(50);
        passwd.setEchoChar('*');

        ok = new Button("OK");

        //생성된 컴포넌트 객체의 배치
        p1.add(llg);
        p1.add(login);
        p2.add(lpw);
        p2.add(passwd);
        p3.add(lpress);
        p3.add(ok);

        p12.add(p1);
        p12.add(p2);

        f.add(p12, "North");
        f.add(p3, "South");

        f.setSize(900,400);
        f.setVisible(true);
    }
```

```java
class LoginUIwithPanelMain
{
    public static void main(String[] args)
    {
        LoginUIwithPanel loginUI = new LoginUIwithPanel("Login Test : HSM");
    }
}
```



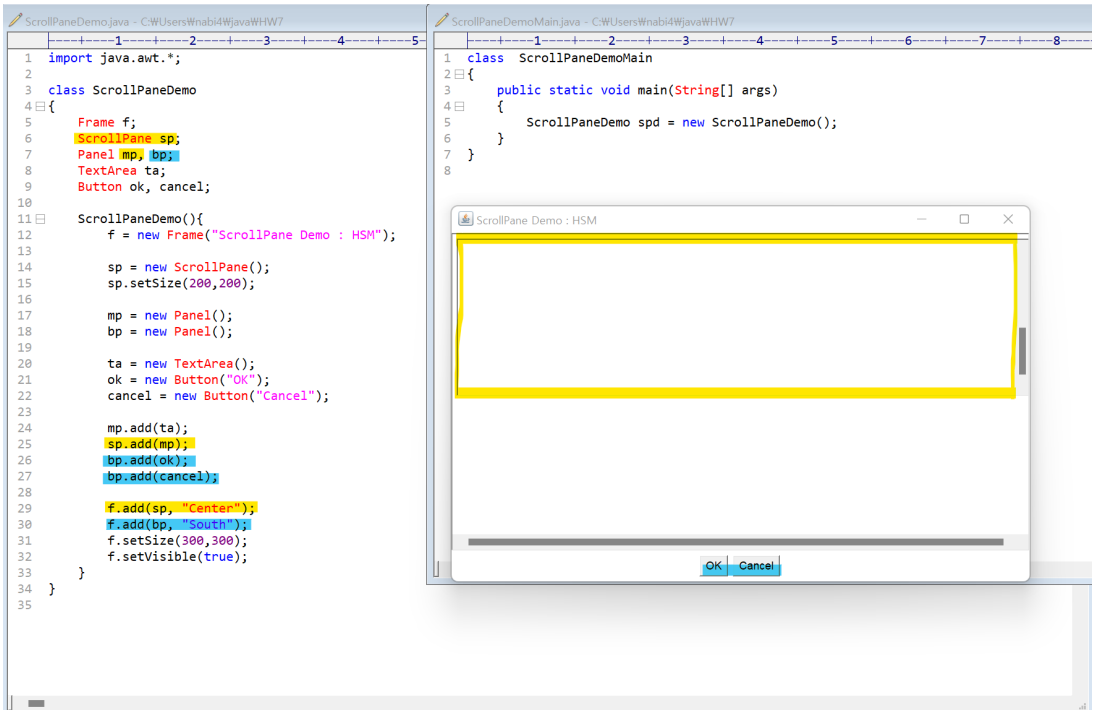
Login과 Password 틀은 위 쪽에, Ok 버튼 틀은 아래 쪽에 위치하여 창을 아무리 변경해도 틀이 안 깨지는 것을 볼 수 있음

### ▼ 4.2.9 ScrollPane

스크롤이 가능한 Panel 기능

#### ▼ Lab1

Panel을 2개를 만들어 첫 번째 패널에는 TextArea를 넣고 이를 ScrollPane으로 감싸 TextArea에 스크롤이 가능하게 함. 두 번째 패널에는 OK, CACNCEL 버튼 2개를 만듬




Panel을 2개를 만들어 첫 번째 패널에는 TextArea를 넣고 이를 ScrollPane으로 감싸 TextArea에 스크롤이 가능하게 함(노란색 부분), 두 번째 패널에는 OK, Cancel 버튼 2개를 만듬(파란색 부분)

### ▼ 4.2.10 PopupMenu

마우스로 클릭했을때 다른 컴포넌트 위에 메뉴를 보여주는 것(container 역할)

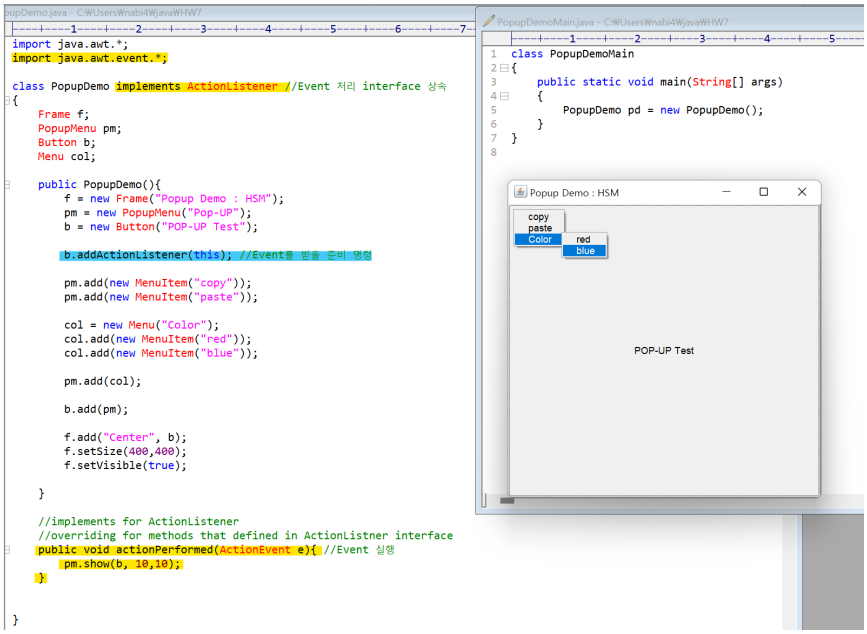
▼ Construct

PopupMenu(), PopupMenu(String label)

▼ Method

add(), show(), hide()

▼ Lab1



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

class PopupDemo implements ActionListener //Event 처리 interface 상속
{
    Frame f;
    PopupMenu pm;
    Button b;
    Menu col;

    public PopupDemo(){
        f = new Frame("Popup Demo : HSM");
        pm = new PopupMenu("Pop-UP");
        b = new Button("POP-UP Test");

        b.addActionListener(this); //Event를 받을 준비 명령

        pm.add(new MenuItem("copy"));
        pm.add(new MenuItem("paste"));

        col = new Menu("Color");
        col.add(new MenuItem("red"));
        col.add(new MenuItem("blue"));

        pm.add(col);

        b.add(pm);

        f.add("Center", b);
        f.setSize(400,400);
        f.setVisible(true);

    }

    //implements for ActionListener
    //overriding for methods that defined in ActionListner interface
    public void actionPerformed(ActionEvent e){ //Event 실행
        pm.show(b, 10,10);
    }


}
```

```
class PopupDemoMain
{
    public static void main(String[] args)
    {
        PopupDemo pd = new PopupDemo();
    }
}
```



PopupMenu만 작성한다고 메뉴창이 뜨는 것은 아님
마우스를 클릭했을때 나타나는 것이므로 이벤트 핸들링이 필요함!!
ActionEvent를 위한 ActionListener를 상속받아 오버라이딩한 부분(노란색) + Button 자체의 메서드를 추가해 이벤트를 받을 준비(파란색)를 모두 해주어야 PopupMenu가 나타남

이때 actionPerformed에서도 쓰이는 변수 pm과 b를 전역변수로 설정해주었기때문에 컴파일 시 에러가 뜨지 않는다. 만약 전역변수가 아닌 public PopupDemo() 안에 지역변수로 설정해준다면 actionPerformed에서는 pm과 b를 찾을 수 없어 에러가 뜰 것이고, 전역변수로 설정해주고 PopupDemo()안에 또 지역변수로도 설정해준다면 actionPerformed에서는 구현이 되지 않은 전역변수 pm과 b를 받기 때문에 popupmenu가 나타나지 않을 것임

## ▼ 4.3 Layout

틀. 컴포넌트들을 어떻게 배치할 것인가에 대한 것

가장 기본적인 것 : BorderLayout(Window, Frame, Dialog), FlowLayout(Panel)

FlowLayout(), BorderLayout(), GridLayout(), CardLayout(), GridBagLayout()

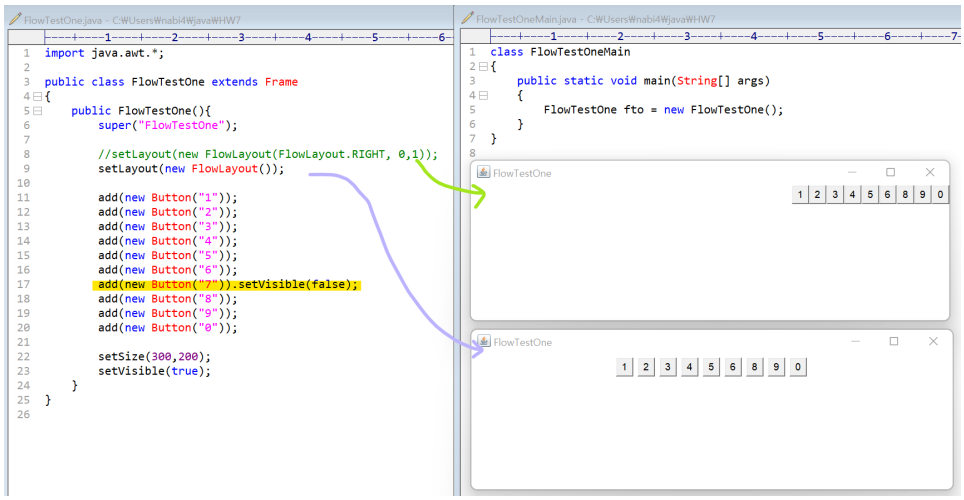
### ▼ 4.3.1 FlowLayout

왼쪽부터 오른쪽까지 쭉 display되는 레이아웃(기본적인 배치 “CENTER”)

▼ Construct

setLayout(new FlowLayout()), setLayout(new FlowLayout(FlowLayout.Right, 10,20))

▼ Lab1



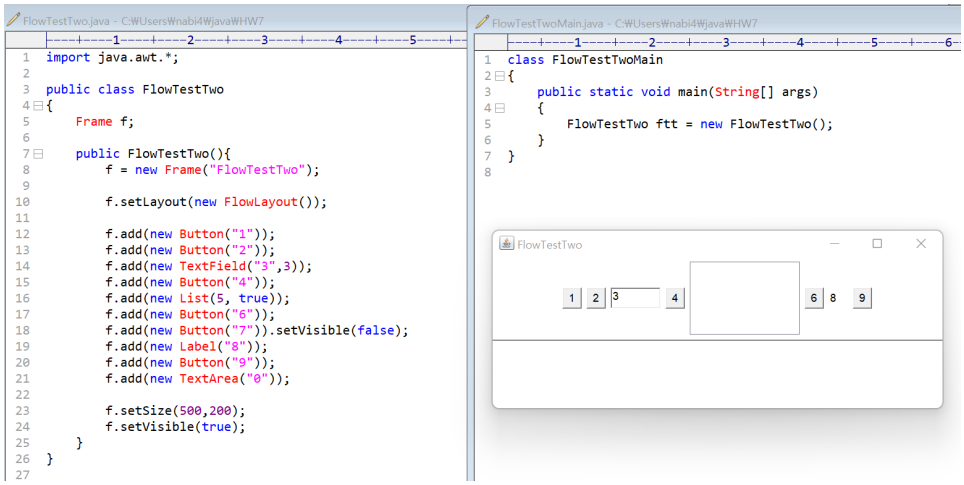
```
import java.awt.*;

public class FlowTestOne extends Frame
{
    public FlowTestOne(){
        super("FlowTestOne");

        //setLayout(new FlowLayout(FlowLayout.RIGHT, 0,1));
        setLayout(new FlowLayout());

        add(new Button("1"));
        add(new Button("2"));
        add(new Button("3"));
        add(new Button("4"));
        add(new Button("5"));
        add(new Button("6"));
        add(new Button("7")).setVisible(false);
        add(new Button("8"));
        add(new Button("9"));
        add(new Button("0"));

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

```
class FlowTestOneMain
{
    public static void main(String[] args)
    {
        FlowTestOne fto = new FlowTestOne();
    }
}
```

setVisible(false)로 설정한 7은 보이지 않음

FlowLayout()은 defalt로 CENTER에 배치되기 때문에 옵션을 주지 않으면 보라색과 같이 중앙에 배치되고, 초록색과 같이 옵션을 주어 배치를 바꿀 수도 있음

▼ Lab2



```
import java.awt.*;

public class FlowTestTwo
{
    Frame f;

    public FlowTestTwo(){
        f = new Frame("FlowTestTwo");

        f.setLayout(new FlowLayout());

        f.add(new Button("1"));
        f.add(new Button("2"));
        f.add(new TextField("3",3));
        f.add(new Button("4"));
        f.add(new List(5, true));
        f.add(new Button("6"));
        f.add(new Button("7")).setVisible(false);
        f.add(new Label("8"));
        f.add(new Button("9"));
        f.add(new TextArea("0"));

        f.setSize(500,200);
        f.setVisible(true);
    }
}
```

```
class FlowTestTwoMain
{
    public static void main(String[] args)
    {
        FlowTestTwo ftt = new FlowTestTwo();
    }
}
```

다양한 컴포넌트들을 넣을 수 있음

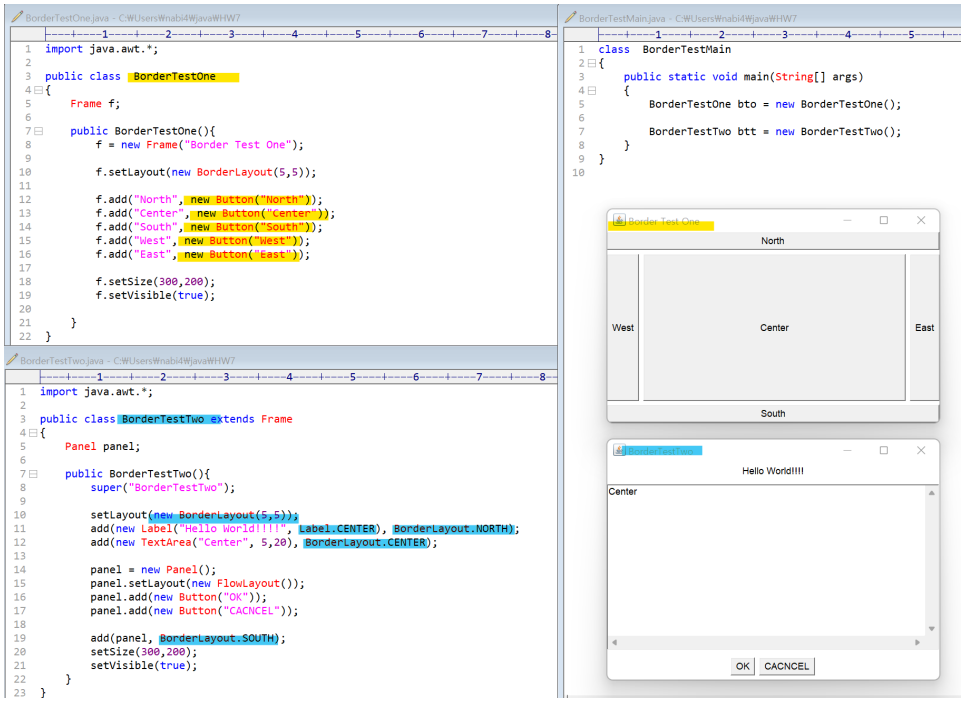
## ▼ 4.3.2 BorderLayout

North, South, East, West, Center

▼ Construct

setLayout(new BorderLayout()), setLayout(new BorderLayout(px,py))

▼ Lab1




BorderLayout에 설정한 위치에 따라 배치가 바뀌는 것을 볼 수 있음

▼ Lab2

앞서 4.2.9 Panel Login 실습에서 실시

## ▼ 4.3.3 GridLayout

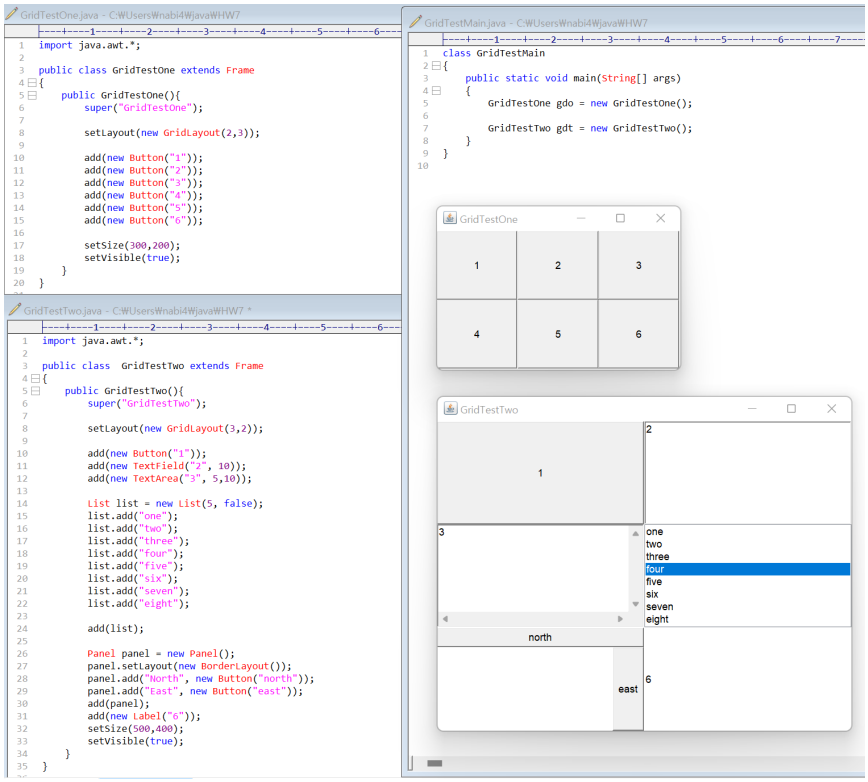
행과 열을 설정할 수 있는 레이아웃

셀의 개수를 지정할 수 있으며 셀에 컴포넌트들을 집어넣을 수 있음

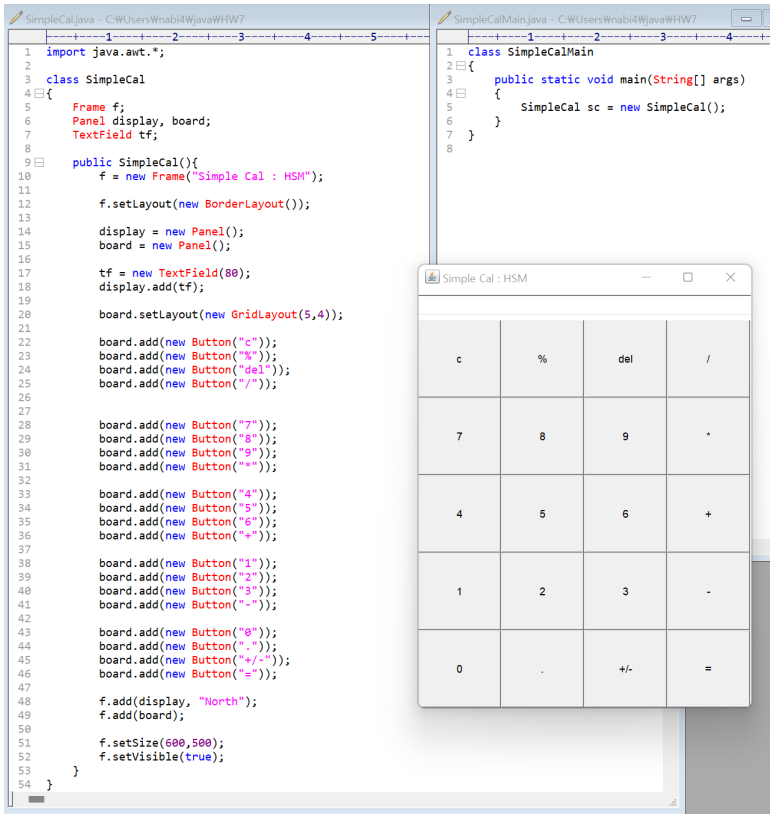
▼ Construct

setLayout(new GridLayout(r,c)), setLayout(new GridLayout(r,c,w,h))

▼ Lab1




▼ Lab3(계산기)




GridLayout을 5행 4열로 한 후 버튼을 추가해주면 다음과 같은 계산기 모양이 만들어짐

▼ 나중 HW (GridBagLayout 참고)

TextField 크기 크게 하기

= 버튼 크기 크게 하기

이벤트 핸들링

### ▼ 4.3.5 GridBagLayout

한 셀의 크기를 확장할 수 있음

Constraint 값을 어떻게 바꾸느냐에 따라 크기가 변경됨

▼ Construct

GridBagLayout gridbag = new GridBagLayout();

▼ Using Flow

1. GridBagLayout gridbag = new GridBagLayout();
2. GridBagConstraint constraint = new GridBagConstraint();
3. setLayout(gridbag);
4. constraint.weightx = 1.0 , constraint.gridheight = 1.0 , constraint.gridwidth = 1.0 ← weightx, wighty, height, width 정의
5. Gridbag.setConstraints(cell, constraint); : setting the constrain to cell
6. add(cell); : add component to container

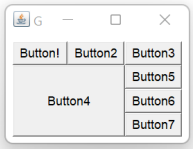
▼ Lab

GridBagTest.java - C:₩Users₩nabi4₩java₩HW7 *

```java
import java.awt.*;

public class GridBagTest extends Frame
{
    public GridBagTest(){
        super("GridBag Test");

        GridBagLayout gridbag = new GridBagLayout();
        GridBagConstraints constraint = new GridBagConstraints();
        setLayout(gridbag);

        constraint.fill = GridBagConstraints.BOTH;
        constraint.weightx = 0.0;
        Button b1 = new Button("Button!");
        gridbag.setConstraints(b1, constraint);
        add(b1);

        Button b2 = new Button("Button2");
        gridbag.setConstraints(b2, constraint);
        add(b2);

        constraint.gridwidth = GridBagConstraints.REMAINDER;
        Button b3 = new Button("Button3");
        gridbag.setConstraints(b3, constraint);
        add(b3);

        constraint.gridwidth = 2;
        constraint.gridheight = 3;
        Button b4 = new Button("Button4");
        gridbag.setConstraints(b4, constraint);
        add(b4);

        constraint.gridwidth = GridBagConstraints.REMAINDER;
        constraint.gridheight = 1;
        constraint.weighty = 0.0;
        Button b5 = new Button("Button5");
        gridbag.setConstraints(b5, constraint);
        add(b5);

        Button b6 = new Button("Button6");
        gridbag.setConstraints(b6, constraint);
        add(b6);

        Button b7 = new Button("Button7");
        gridbag.setConstraints(b7, constraint);
        add(b7);

        pack();
        setVisible(true);
    }
}
```

GridBagTestMain.java - C:₩Users₩nabi4₩java₩HW7

```java
class GridBagTestMain
{
    public static void main(String[] args)
    {
        GridBagTest gbt = new GridBagTest();
    }
}
```




gridwidht와 gridheight의 크기를 다르게 주면 컴포넌트의 크기를 변경 가능

## ▼ HW(계산기_ TextField 크기 크게 하기, 버튼 크기 크게 하기)

GridBagLayoutCalHW.java - C:₩Users₩nabi4₩java₩HW7

```java
import java.awt.*;

public class GridBagLayoutCalHW
{
    Frame f;
    TextField tf;

    public  GridBagLayoutCalHW(){
        f = new Frame("GridBagLayoutCalHW");
        GridBagLayout gridbag = new GridBagLayout();
        GridBagConstraints constraint = new GridBagConstraints();
        f.setLayout(gridbag);

        constraint.fill = GridBagConstraints.BOTH;
        constraint.weightx = 1.0;
        constraint.weighty = 1.5;
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        constraint.gridheight = 2;
        tf = new TextField(100);
        f.add(tf, constraint);

        constraint.gridwidth = 1;
        constraint.gridheight = 1;
        f.add(new Button("C"), constraint);
        f.add(new Button("CE"), constraint);
        f.add(new Button("1/x"), constraint);
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("DEL"), constraint);

        constraint.gridwidth = 1;
        f.add(new Button("%"), constraint);
        f.add(new Button("/"), constraint);
        f.add(new Button("X"), constraint);
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("-"), constraint);

        constraint.gridwidth = 1;
        f.add(new Button("7"), constraint);
        f.add(new Button("8"), constraint);
        f.add(new Button("9"), constraint);
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("+"), constraint);

        constraint.gridwidth = 1;
        f.add(new Button("4"), constraint);
        f.add(new Button("5"), constraint);
        f.add(new Button("6"), constraint);
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("()"), constraint);

        constraint.gridwidth = 1;
        f.add(new Button("1"), constraint);
        f.add(new Button("2"), constraint);
        f.add(new Button("3"), constraint);
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("+/-"), constraint);

        constraint.gridwidth = 1;
        f.add(new Button("0"), constraint);
        f.add(new Button("."), constraint);
        constraint.gridwidth = 2;
        constraint.gridwidth = GridBagConstraints.REMAINDER;
        f.add(new Button("="), constraint);

        f.setSize(400,600);
        f.setVisible(true);


    }
}
```

GridBagLayoutCalHWMain.java - C:₩Users₩nabi4₩java₩HW7

```java
class GridBagLayoutCalHWMain
{
    public static void main(String[] args)
    {
        GridBagLayoutCalHW cal = new GridBagLayoutCalHW();
    }
}
```

위에서 살펴본 예제를 인용하여 weightx,weighty, gridwidth, gridheight 값을 변경해 textfield 크기도 크게 하고 “=”버튼의 크기를 크게 만들었음


GridBagLayoutCalHW
C
CE
1/x
DEL
%
/
X
-
7
8
9
+
4
5
6
()
1
2
3
+/-
0
.
=