Layout

1. 請參考投影片內容，建立以下視窗應用程式

請將1.程式**執行結果**截圖置入作業中、2.程式原始檔置入作業中

1. 使用Flow Layout建立一基本的視窗程式

import java.awt.\*;

import java.awt.event.\*;

public class FlowLayoutDemo extends java.awt.Frame {

public static void main(String args[]){

new FlowLayoutDemo();

}

// 建構函式

public FlowLayoutDemo() {

super("Flow Layout Demo");

// 向水平中央對齊

FlowLayout flowlayout = new FlowLayout(FlowLayout.CENTER);

// 設定物件間的水平間距

flowlayout.setHgap(10);

// 設定物件間的垂直間距

flowlayout.setVgap(10);

// 設定Layout Manager

setLayout(flowlayout);

Button button1 = new Button("OK");

Button button2 = new Button("Cancel");

Button button3 = new Button("Yes");

Button button4 = new Button("No");

// 將物件加至Frame中

add(button1);

add(button2);

add(button3);

add(button4);

// 設定視窗的大小

this.setSize(300, 100);

// Center the frame

Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

Dimension frameSize = this.getSize();

if (frameSize.height > screenSize.height)

frameSize.height = screenSize.height;

if (frameSize.width > screenSize.width)

frameSize.width = screenSize.width;

this.setLocation((screenSize.width - frameSize.width) / 2, (screenSize.height - frameSize.height) / 2);

// 顯示視窗

this.setVisible(true);

this.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

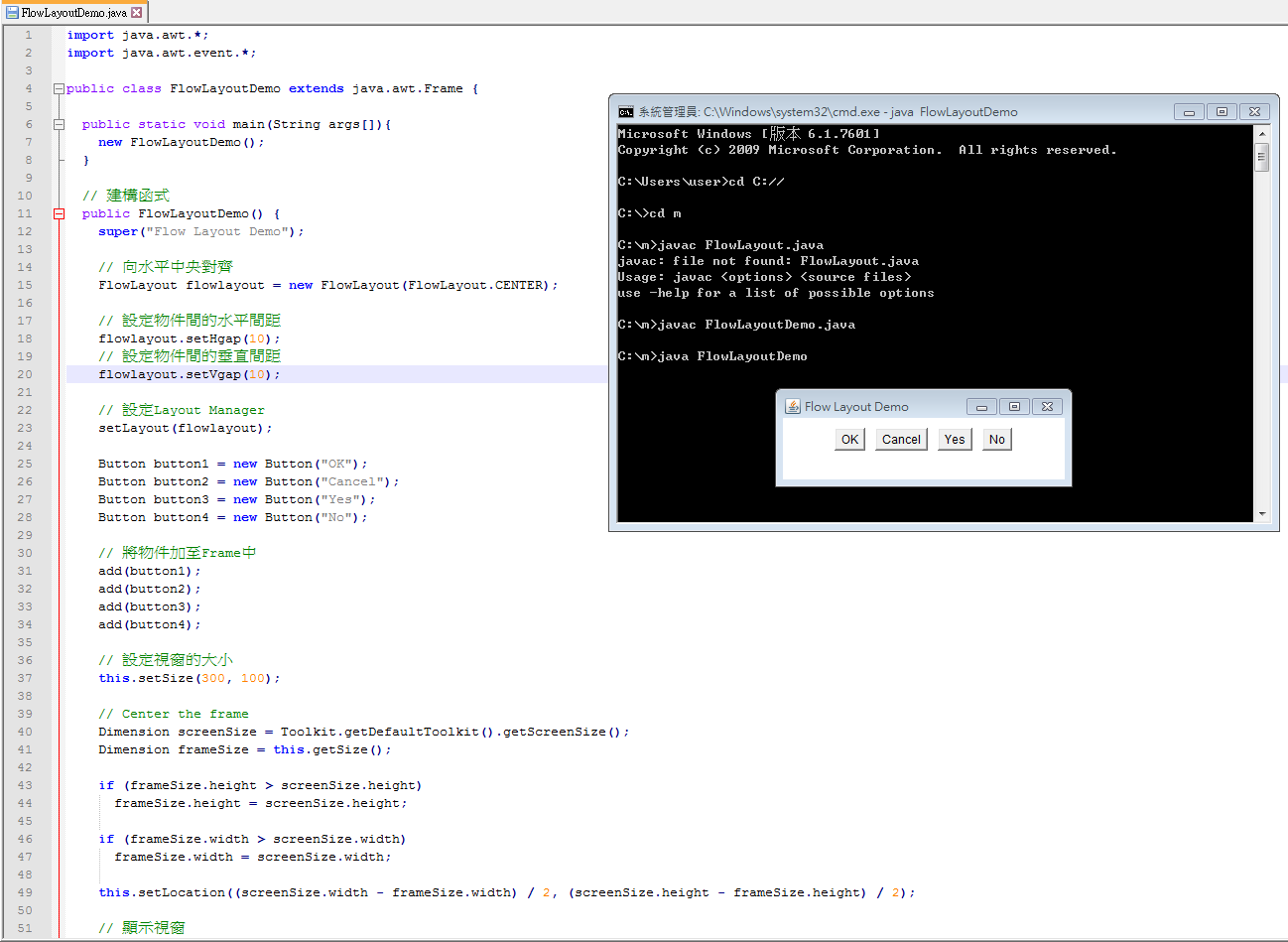
System.exit(0);

}

});

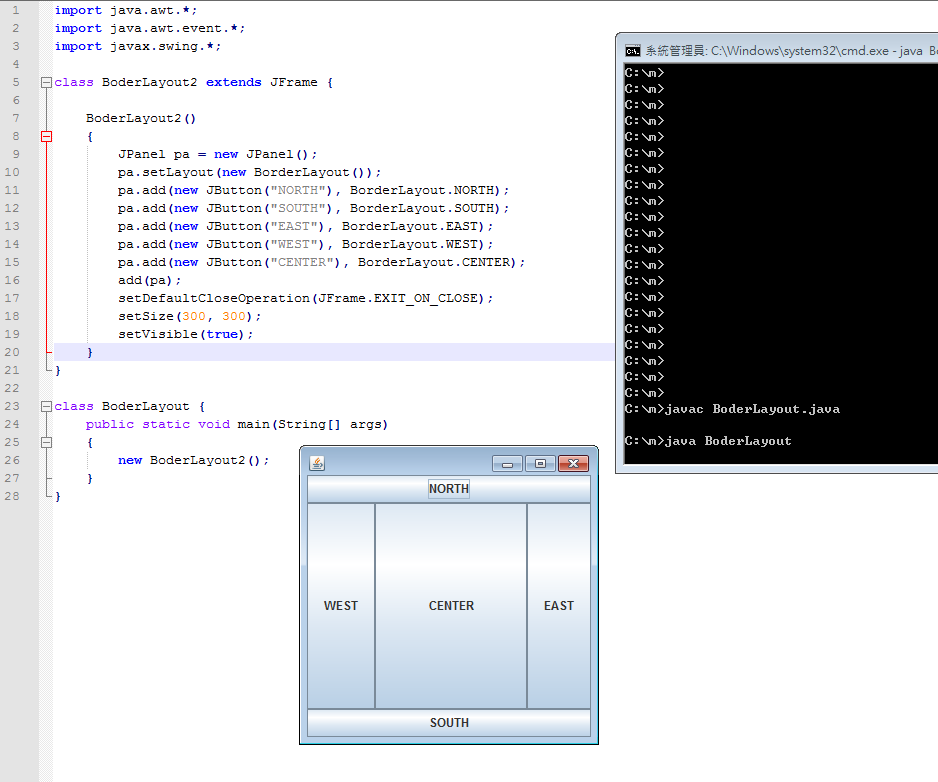
}

}



1. 使用Boder Layout建立一基本的視窗程式，結果如下圖





import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

class BoderLayoutDemo extends JFrame {

BoderLayoutDemo()

{

JPanel pa = new JPanel();

pa.setLayout(new BorderLayout());

pa.add(new JButton("NORTH"), BorderLayout.NORTH);

pa.add(new JButton("SOUTH"), BorderLayout.SOUTH);

pa.add(new JButton("EAST"), BorderLayout.EAST);

pa.add(new JButton("WEST"), BorderLayout.WEST);

pa.add(new JButton("CENTER"), BorderLayout.CENTER);

add(pa);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(300, 300);

setVisible(true);

}

}

class MainFrame {

public static void main(String[] args)

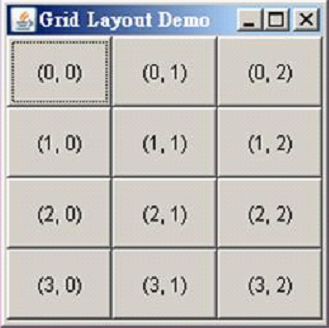
{

new BoderLayoutDemo();

}

}

1. 使用Grid Layout建立一基本的視窗程式，結果如下圖



import java.awt.\*;

import java.awt.event.\*;

public class MyGridLayout extends java.awt.Frame {

public static void main(String[] args) {

new MyGridLayout();

}

public MyGridLayout() {

Frame frame = new Frame("Grid Layout");

frame.addWindowListener(new AdapterDemo());

frame.setLayout(new GridLayout(4, 3));

Button b1 = new Button("0,0");

Button b2 = new Button("0,1");

Button b3 = new Button("0,2");

Button b4 = new Button("1,0");

Button b5 = new Button("1,1");

Button b6 = new Button("1,2");

Button b7 = new Button("2,0");

Button b8 = new Button("2,1");

Button b9 = new Button("2,2");

Button b10 = new Button("3,0");

Button b11= new Button("3,1");

Button b12 = new Button("3,2");

this.setSize(300, 300);

frame.add(b1);

frame.add(b2);

frame.add(b3);

frame.add(b4);

frame.add(b5);

frame.add(b6);

frame.add(b7);

frame.add(b8);

frame.add(b9);

frame.add(b10);

frame.add(b11);

frame.add(b12);

this.setSize(300, 300);

frame.pack();

frame.setVisible(true);

}

}

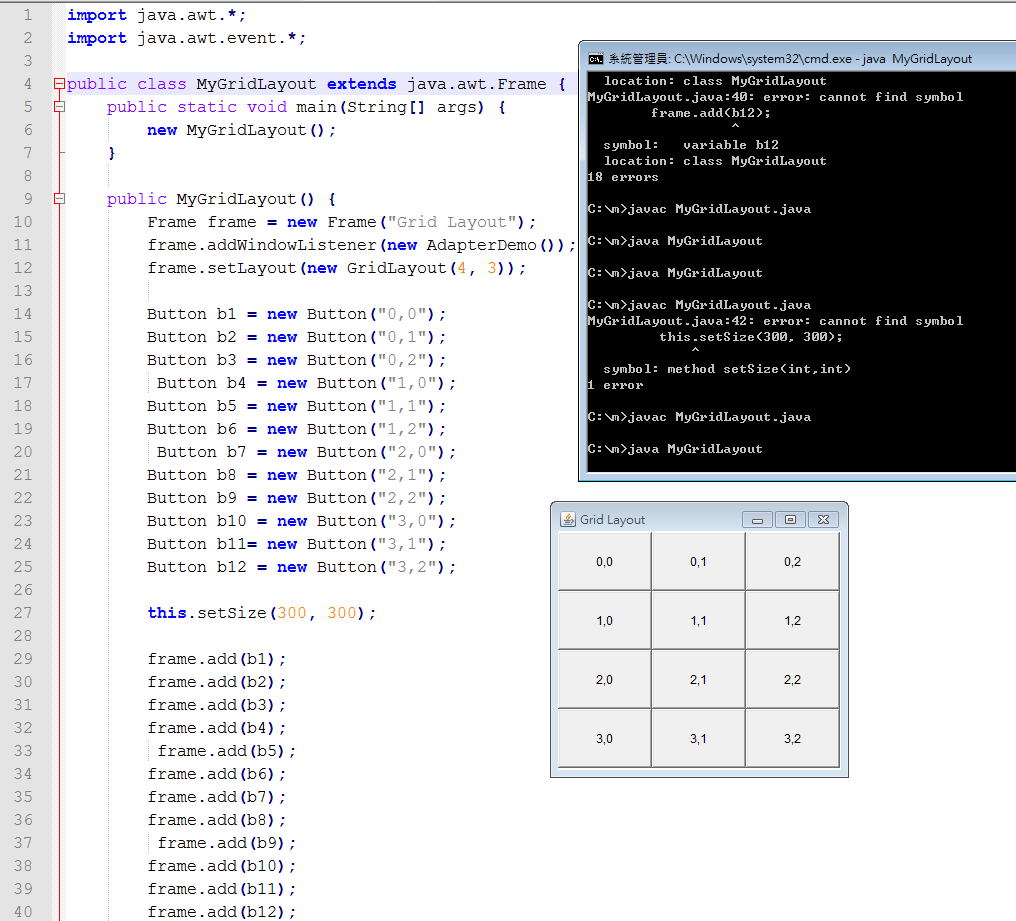
class AdapterDemo extends WindowAdapter {

public void windowClosing(WindowEvent e) {

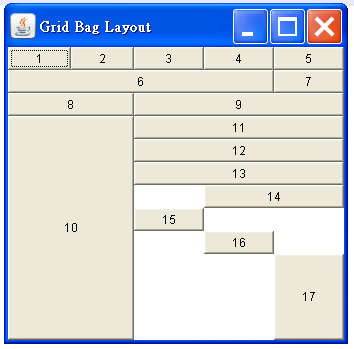
System.exit(0);

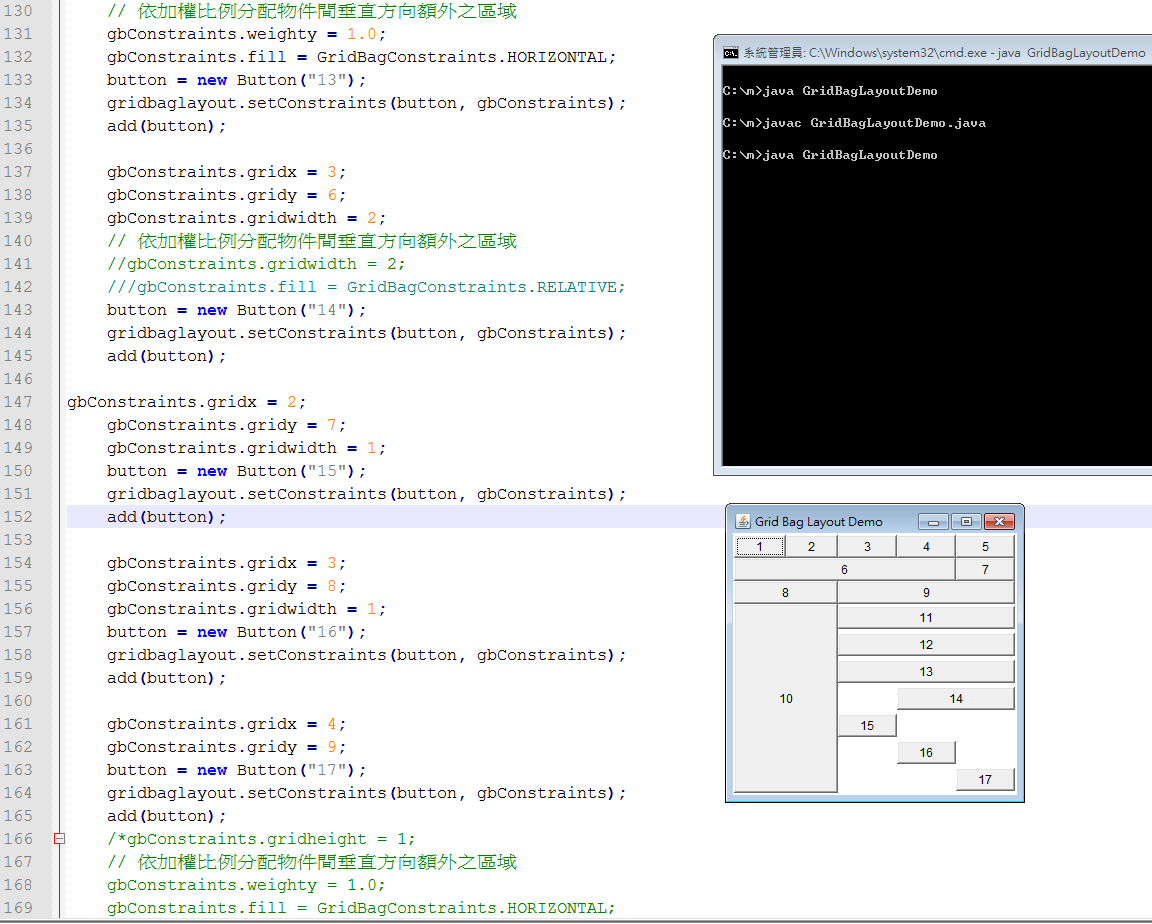
}

}



1. 使用Grid Bag Layout建立一基本的視窗程式，結果如下圖





import java.awt.\*;

import java.awt.event.\*;

public class GridBagLayoutDemo extends java.awt.Frame {

public static void main(String args[]){

new GridBagLayoutDemo();

}

// 建構函式

public GridBagLayoutDemo() {

super("Grid Bag Layout Demo");

Button button;

GridBagLayout gridbaglayout = new GridBagLayout();

GridBagConstraints gbConstraints = new GridBagConstraints();

// 定義 Layout Manager 為 GridBagLayout

setLayout(gridbaglayout);

// 同時改變物件寬度與高度以填滿顯示區域之水平與垂直方向

gbConstraints.fill = GridBagConstraints.BOTH;

// 依加權比例分配物件間水平方向額外之區域

gbConstraints.weightx = 0.5;

button = new Button("1");

// 設定Grid Bag Layout中物件的限制

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

button = new Button("2");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

// 設定物件配置時所佔據區域列的數目

// 物件將填滿所剩餘的區域列數或行數

button = new Button("3");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

// 依加權比例分配物件間水平方向額外之區域

button = new Button("4");

gridbaglayout.setConstraints(button, gbConstraints);

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

add(button);

// 設定物件配置時所佔據區域列的數目

// 物件將填滿除了所在列的最後一個單位區域以外之區域列數

button = new Button("5");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

// 設定物件配置時所佔據區域列的數目

// 物件將填滿所剩餘的區域列數或行數

gbConstraints.fill = GridBagConstraints.BOTH;

gbConstraints.gridwidth = 4;

// gbConstraints.weighty = 1.0;

button = new Button("6");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

// 設定物件配置時所佔據區域列的數目

//gbConstraints.gridwidth = 1;

// 設定物件配置時所佔據區域行的數目

// gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

//gbConstraints.weighty = 1.0;

button = new Button("7");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

//////////

gbConstraints.fill = GridBagConstraints.BOTH;

gbConstraints.gridwidth = 2;

// gbConstraints.weighty = 1.0;

button = new Button("8");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

// 設定物件配置時所佔據區域列的數目

//gbConstraints.gridwidth = 1;

// 設定物件配置時所佔據區域行的數目

// gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

//gbConstraints.weighty = 1.0;

button = new Button("9");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

// 依加權比例分配物件間垂直方向額外之區域

// 設定物件配置時所佔據區域列的數目

gbConstraints.gridwidth = 2;

// 設定物件配置時所佔據區域行的數目

gbConstraints.gridheight = 7;

// 依加權比例分配物件間垂直方向額外之區域

gbConstraints.weighty = 1.0;

button = new Button("10");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

// 設定物件配置時所佔據區域行的數目

gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

gbConstraints.weighty = 1.0;

gbConstraints.fill = GridBagConstraints.HORIZONTAL;

button = new Button("11");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

// 設定物件配置時所佔據區域行的數目

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

// 設定物件配置時所佔據區域行的數目

gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

gbConstraints.weighty = 1.0;

gbConstraints.fill = GridBagConstraints.HORIZONTAL;

button = new Button("12");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridwidth = GridBagConstraints.REMAINDER;

gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

gbConstraints.weighty = 1.0;

gbConstraints.fill = GridBagConstraints.HORIZONTAL;

button = new Button("13");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridx = 3;

gbConstraints.gridy = 6;

gbConstraints.gridwidth = 2;

// 依加權比例分配物件間垂直方向額外之區域

//gbConstraints.gridwidth = 2;

///gbConstraints.fill = GridBagConstraints.RELATIVE;

button = new Button("14");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridx = 2;

gbConstraints.gridy = 7;

gbConstraints.gridwidth = 1;

button = new Button("15");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridx = 3;

gbConstraints.gridy = 8;

gbConstraints.gridwidth = 1;

button = new Button("16");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

gbConstraints.gridx = 4;

gbConstraints.gridy = 9;

button = new Button("17");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

/\*gbConstraints.gridheight = 1;

// 依加權比例分配物件間垂直方向額外之區域

gbConstraints.weighty = 1.0;

gbConstraints.fill = GridBagConstraints.HORIZONTAL;

button = new Button("15");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

\*/

/\*button = new Button("13");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

button = new Button("14");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

button = new Button("15");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

button = new Button("16");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

button = new Button("17");

gridbaglayout.setConstraints(button, gbConstraints);

add(button);

\*/

// 設定視窗的大小

this.setSize(300, 300);

// Center the frame

Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

Dimension frameSize = this.getSize();

if (frameSize.height > screenSize.height)

frameSize.height = screenSize.height;

if (frameSize.width > screenSize.width)

frameSize.width = screenSize.width;

this.setLocation((screenSize.width - frameSize.width) / 2, (screenSize.height - frameSize.height) / 2);

// 顯示視窗

this.setVisible(true);

this.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

System.exit(0);

}

});

}

}