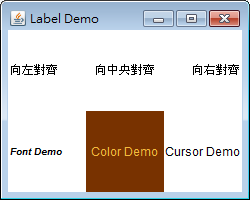
Label & Button

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. 建立6個各種樣式的Label
   1. 向左對齊的Label
   2. 向水平中央對齊的Label
   3. 向右對齊的Label
   4. 具有字型的Label
   5. 具有前景顏色與背景顏色的Label
   6. 具有不同滑鼠指標的Label



import java.awt.\*;

import java.awt.event.\*;

public class LabelDemo extends java.awt.Frame {

public static void main(String args[]){

new LabelDemo();

}

// 建構函式

public LabelDemo() {

super("Label Demo");

final int row = 2; // 列

final int column = 3; // 行

// 定義 Layout Manager 為 GridLayout

setLayout(new GridLayout(row, column));

java.awt.Label label;

// 建構函式 1

label = new Label();

// 設定Label類別之顯示文字

label.setText("向左對齊");

// 設定Label類別之對齊方式，向左對齊

label.setAlignment(Label.LEFT);

add(label);

// 建構函式 2

label = new Label("向中央對齊");

// 設定Label類別之對齊方式，向中央對齊

label.setAlignment(Label.CENTER);

add(label);

// 建構函式 3

// 設定Label類別之對齊方式，向右對齊

label = new Label("向右對齊", Label.RIGHT);

add(label);

// 使用Component抽象類別所提供的方法

// 設定字型 - 粗斜體字型

label = new Label("Font Demo", Label.LEFT);

label.setFont(new Font("dialog", Font.BOLD | Font.ITALIC, 10));

add(label);

// 設定顏色

label = new Label("Color Demo", Label.CENTER);

// 設定背景顏色

label.setBackground(new Color(120,50,0));

// 設定前景顏色

label.setForeground(new Color(245,185,60));

add(label);

// 設定滑鼠指標

label = new Label("Cursor Demo", Label.RIGHT);

// 選擇連線（手形）之滑鼠指標

label.setCursor(new Cursor(Cursor.HAND\_CURSOR));

add(label);

// 設定視窗的大小

this.setSize(250, 200);

// 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);

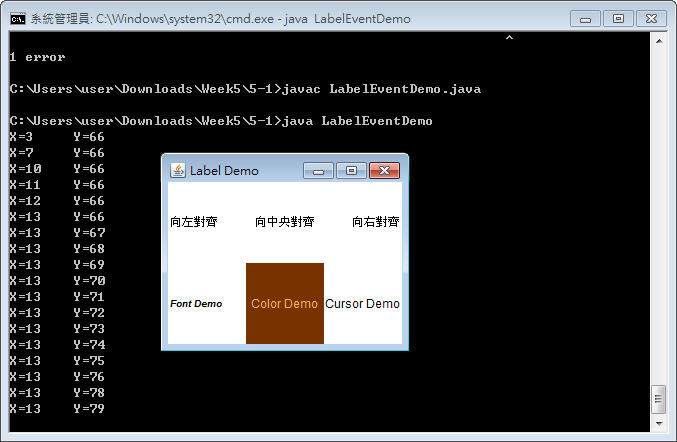
}

});

}

}

1. 建立1個Label，並使其具有以下監控事件(只有這個Label有,其他區域都沒有), **執行結果可參考moodle**
   1. Mouse
   2. Mouse Motion Listener



import java.awt.\*;

import java.awt.event.\*;

public class LabelEventDemo extends java.awt.Frame{

public static void main(String args[]){

new LabelEventDemo();

}

// 建構函式

public LabelEventDemo() {

super("Label Demo");

final int row = 2; // 列

final int column = 3; // 行

// 定義 Layout Manager 為 GridLayout

setLayout(new GridLayout(row, column));

java.awt.Label label;

// 建構函式 1

label = new Label();

// 設定Label類別之顯示文字

label.setText("向左對齊");

// 設定Label類別之對齊方式，向左對齊

label.setAlignment(Label.LEFT);

add(label);

// 建構函式 2

label = new Label("向中央對齊");

// 設定Label類別之對齊方式，向中央對齊

label.setAlignment(Label.CENTER);

add(label);

label.addMouseMotionListener(new MouseMotionListener(){

public void mouseMoved( MouseEvent e)

{

System.out.println("X=" + e.getX() + "\tY=" + e.getY());

}

public void mouseDragged( MouseEvent e)

{

System.out.println("X=" + e.getX() + "\tY=" + e.getY());

}

});

// 建構函式 3

// 設定Label類別之對齊方式，向右對齊

label = new Label("向右對齊", Label.RIGHT);

add(label);

// 使用Component抽象類別所提供的方法

// 設定字型 - 粗斜體字型

label = new Label("Font Demo", Label.LEFT);

label.setFont(new Font("dialog", Font.BOLD | Font.ITALIC, 10));

add(label);

// 設定顏色

label = new Label("Color Demo", Label.CENTER);

// 設定背景顏色

label.setBackground(new Color(120,50,0));

// 設定前景顏色

label.setForeground(new Color(245,185,60));

add(label);

// 設定滑鼠指標

label = new Label("Cursor Demo", Label.RIGHT);

// 選擇連線（手形）之滑鼠指標

label.setCursor(new Cursor(Cursor.HAND\_CURSOR));

add(label);

// 設定視窗的大小

this.setSize(250, 200);

// 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);

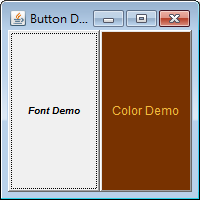
}

});

}

}

1. 建立2個各種樣式的Button
   1. 具有字型的Button
   2. 具有前景顏色與背景顏色的Button



import java.awt.\*;

import java.awt.event.\*;

public class ButtonDemo extends java.awt.Frame {

public static void main(String args[]){

new ButtonDemo();

}

// 建構函式

public ButtonDemo() {

super("Button Demo");

final int row = 1; // 列

final int column = 2; // 行

// 定義 Layout Manager 為 GridLayout

setLayout(new GridLayout(row, column));

java.awt.Button button;

// 設定字型 - 粗斜體字型

button = new Button("Font Demo");

button.setFont(new Font("dialog", Font.BOLD | Font.ITALIC, 10));

add(button);

// 設定顏色

button = new Button("Color Demo");

// 設定背景顏色

button.setBackground(new Color(120,50,0));

// 設定前景顏色

button.setForeground(new Color(245,185,60));

add(button);

// 設定視窗的大小

this.setSize(200, 200);

// 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);

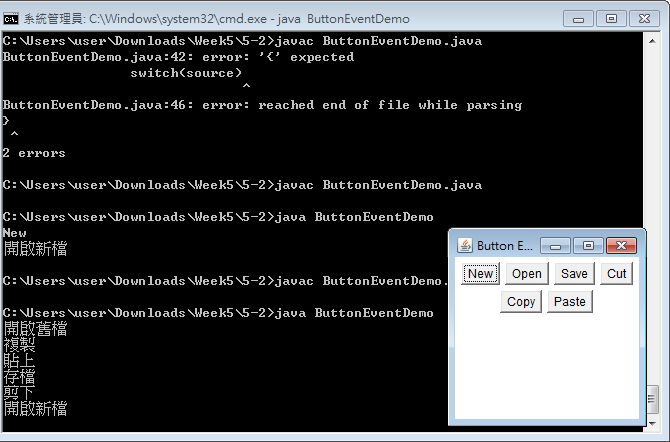
}

});

}

}

1. 建立6個Button，並使其具有動作事件並有相關動作ㄝ**執行結果可參考moodle**
   1. New，被按下時印出 開啟新檔
   2. Open，被按下時印出 開啟舊檔
   3. Save，被按下時印出 存檔
   4. Cut，被按下時印出 剪下
   5. Copy，被按下時印出 複製
   6. Paste，被按下時印出 貼上



import java.awt.\*;

import java.awt.event.\*;

public class ButtonEventDemo extends java.awt.Frame implements ActionListener {

Button[] button = new Button[6];

public static void main(String[] args){

new ButtonEventDemo();

}

public ButtonEventDemo()

{

super("Button Event Demo");

setLayout(new FlowLayout());

String[] label = {"New", "Open", "Save", "Cut", "Copy", "Paste"};

for (int i=0; i<6; i++)

{

button[i] = new Button(label[i]);

button[i].addActionListener(this);

add(button[i]);

}

this.setSize(200,200);

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);

this.setVisible(true);

this.addWindowListener(new WindowAdapter()

{

public void windowClossing(WindowEvent e)

{

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent e)

{

String source = e.getActionCommand();

switch(source){

case "New":

System.out.println("開啟新檔");

break;

case "Open":

System.out.println("開啟舊檔");

break;

case "Save":

System.out.println("存檔");

break;

case "Cut":

System.out.println("剪下");

break;

case "Copy":

System.out.println("複製");

break;

case "Paste":

System.out.println("貼上");

break;

}

}

}