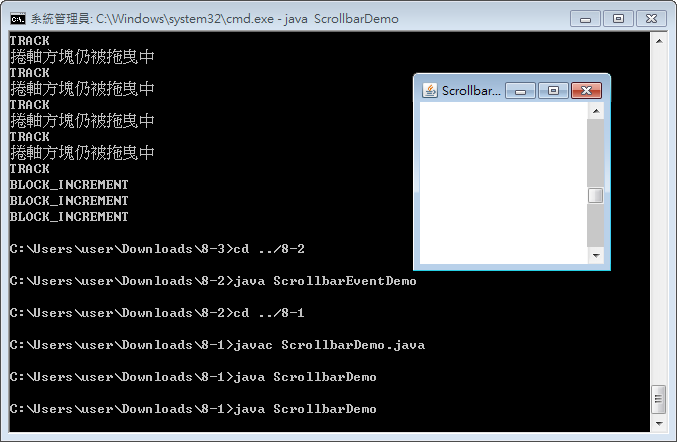
Scrollbar

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. 建立具有Scrollbar的視窗應用程式,Scrollbar的屬性如下
   1. 最大值為500, 最小值為0
   2. 垂直捲軸
   3. 捲軸方塊的大小為10
   4. 捲軸之目前值為300
   5. 設定Block遞增或遞減數目為15
   6. 設定Unit遞增或遞減數目為3



import java.awt.\*;

import java.awt.event.\*;

public class ScrollbarDemo extends java.awt.Frame {

public static void main(String args[]){

new ScrollbarDemo();

}

// 建構函式

public ScrollbarDemo() {

super("Scrollbar Demo");

// 定義 Layout Manager 為 BorderLayout

setLayout(new BorderLayout());

java.awt.Scrollbar scrollbar;

// 建構函式 3

// 設定配置方向為垂直捲軸

// 目前值為100

// 捲軸方塊大小值為60

// 最小值為0

// 最大值為300

scrollbar = new Scrollbar(Scrollbar.VERTICAL, 300, 10, 0, 500);

scrollbar.setUnitIncrement(3);

scrollbar.setBlockIncrement(15);

add(scrollbar, BorderLayout.EAST);

// 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);

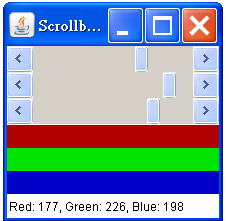
}

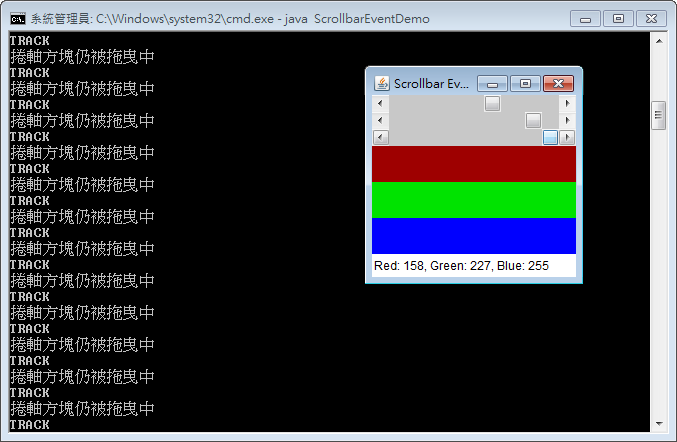
});

}

}

1. 建立具有Adjustment Event的視窗應用程式,
   1. 建立3個捲軸,分別可設定R,G,B三元色的值
   2. 建立3個panel,分別顯示R,G,B
   3. 調整R的捲軸時可立即顯示R的panel , 由亮紅到暗紅 , 綠色與黑色也是
   4. 例子8-2是將RGB全部顯示在同一個panel,請試著將這三個全部分開顯示
   5. 如下圖所示,也可參考moodle上所放的執行結果





import java.awt.\*;

import java.awt.event.\*;

public class ScrollbarEventDemo extends java.awt.Frame implements AdjustmentListener {

Scrollbar scrollbar[] = new Scrollbar[3] ;

Panel colorPanel1 = new Panel();

Panel colorPanel2 = new Panel();

Panel colorPanel3 = new Panel();

Label label = new Label();

int red = 0;

int green = 0;

int blue = 0;

public static void main(String args[]){

new ScrollbarEventDemo();

}

// 建構函式

public ScrollbarEventDemo() {

super("Scrollbar Event Demo");

// 定義 Layout Manager 為 BorderLayout

setLayout(new BorderLayout());

Panel panel = new Panel();

panel.setLayout(new GridLayout(3, 1));

Panel colorPanel = new Panel();

colorPanel.setLayout(new GridLayout(3,1));

for (int i=0; i < 3; i++) {

// 設定配置方向為水平捲軸

// 目前值為0, 捲軸方塊大小值為0

// 最小值為0, 最大值為256

scrollbar[i] = new Scrollbar(Scrollbar.HORIZONTAL, 0, 0, 0, 256);

// 註冊 AdjustmentListener

scrollbar[i].addAdjustmentListener(this);

panel.add(scrollbar[i]);

}

label.setText("Red: " + red + ", Green: " + green + ", Blue: " + blue);

colorPanel1.setBackground(new Color(red, 0, 0));

colorPanel2.setBackground(new Color(0, green, 0));

colorPanel3.setBackground(new Color(0, 0, blue));

colorPanel.add(colorPanel1);

colorPanel.add(colorPanel2);

colorPanel.add(colorPanel3);

this.add(panel, BorderLayout.NORTH);

this.add(colorPanel, BorderLayout.CENTER);

this.add(label, BorderLayout.SOUTH);

// 設定視窗的大小

this.setSize(220, 220);

// 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);

}

});

}

public void adjustmentValueChanged(AdjustmentEvent e) {

// 取得產生調校事件之物件

Scrollbar sb = (Scrollbar)(e.getAdjustable());

// 取得產生調校事件時捲軸之目前值

if (sb == scrollbar[0])

red = sb.getValue();

else if (sb == scrollbar[1])

green = sb.getValue();

else if (sb == scrollbar[2])

blue = sb.getValue();

label.setText("Red: " + red + ", Green: " + green + ", Blue: " + blue);

colorPanel1.setBackground(new Color(red, 0, 0));

colorPanel2.setBackground(new Color(0, green, 0));

colorPanel3.setBackground(new Color(0, 0, blue));

// 回傳產生調校事件時之類型

int type = e.getAdjustmentType();

switch (type) {

case AdjustmentEvent.BLOCK\_DECREMENT:

System.out.println("BLOCK\_DECREMENT");

break;

case AdjustmentEvent.BLOCK\_INCREMENT:

System.out.println("BLOCK\_INCREMENT");

break;

case AdjustmentEvent.UNIT\_DECREMENT:

System.out.println("UNIT\_DECREMENT");

break;

case AdjustmentEvent.UNIT\_INCREMENT:

System.out.println("UNIT\_INCREMENT");

break;

case AdjustmentEvent.TRACK:

System.out.println("TRACK");

break;

default:

break;

}

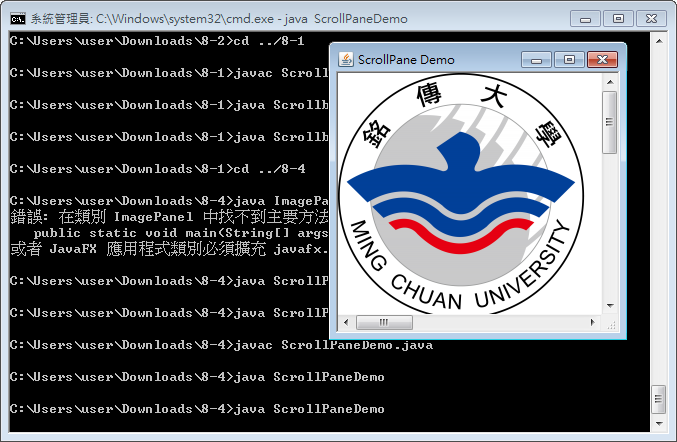
if (e.getValueIsAdjusting())

System.out.println("捲軸方塊仍被拖曳中");

}

}

1. 建立具有捲軸面版的視窗應用程式,
   1. 捲軸面版中放入與學校相關的圖面
   2. 捲軸為隨時顯示水平與垂直捲軸
   3. 設定支援滑鼠滾輪
   4. 尺寸為1024x768



import java.awt.\*;

import java.awt.event.\*;

public class ScrollPaneDemo extends java.awt.Frame {

// Main method

public static void main(String[] args) {

new ScrollPaneDemo();

}

// 建構函式

public ScrollPaneDemo() {

super("ScrollPane Demo");

// 定義 Layout Manager 為 BorderLayout

setLayout(new BorderLayout());

// 捲軸面板

ScrollPane scrollpane = new ScrollPane(ScrollPane.SCROLLBARS\_ALWAYS);

// 支援滑鼠滾輪功能

scrollpane.setWheelScrollingEnabled(true);

// 自訂類別

ImagePanel imagepanel = new ImagePanel();

// 設定內置物件之最佳尺寸

imagepanel.setPreferredSize(new Dimension(1024, 768));

scrollpane.add(imagepanel);

this.add(scrollpane, BorderLayout.CENTER);

// 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);

}

});

}

}

class ImagePanel extends java.awt.Panel {

Image image;

// 建構函式

public ImagePanel() {

ClassLoader cl = this.getClass().getClassLoader();

Toolkit tk = Toolkit.getDefaultToolkit();

// 取得圖像

image = tk.createImage(cl.getResource("images/logo-img.png"));

}

public void paint(Graphics g) {

// 繪製圖像

g.drawImage(image, 0, 0, this);

}

public void update(Graphics g) {

paint(g);

}

}