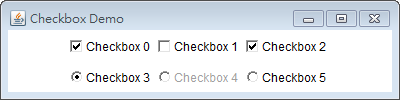
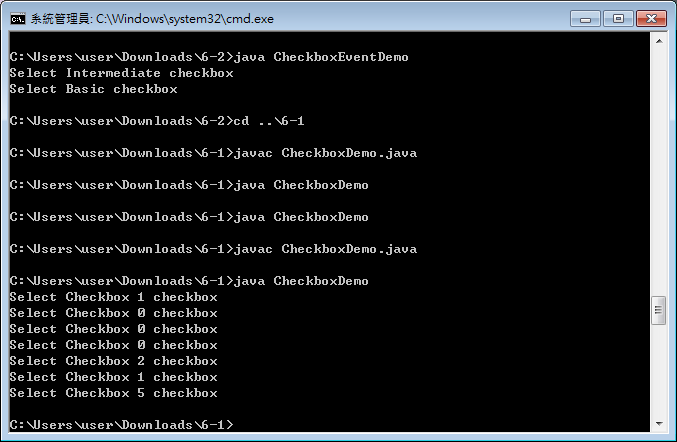
Checkbox, List, Choice

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. 建立各種樣式的Checkbox
   1. 使用建構式 public Checkbox()建立，顯示文字為Checkbox0、並設定核取狀態為選取
   2. 使用建構式 public Checkbox(String label)建立，顯示文字為Checkbox1、並設定核取狀態為未選取
   3. 使用建構式 public Checkbox(String label, boolean state)建立，顯示文字為Checkbox2、並設定核取狀態為選取
   4. 建立CheckboxGroup 內有 3個 Checkbox，分別為
      1. 使用建構式 public Checkbox()建立，顯示文字為Checkbox3、並設定核取狀態為選取
      2. 使用建構式 public Checkbox(String label, boolean state, CheckboxGroup group)建立，顯示文字為Checkbox4、並設定Enabled為false
      3. 使用建構式 public Checkbox(String label, CheckboxGroup group ,boolean state)建立，顯示文字為Checkbox5
      4. 並請針對上述3個Checkbox加上Item Event Listener，點選Checkbox產生事件動作並列印出該Checkbox的label文字





import java.awt.\*;

import java.awt.event.\*;

public class CheckboxDemo extends java.awt.Frame implements ItemListener {

java.awt.Checkbox[] checkbox = new Checkbox[6];

public static void main(String args[]){

new CheckboxDemo();

}

// 建構函式

public CheckboxDemo() {

super("Checkbox Demo");

final int row = 2; // 列

final int column = 1; // 行

// 定義 Layout Manager 為 GridLayout

setLayout(new GridLayout(row, column));

java.awt.Panel panel1 = new Panel();

// 建構函式 1

checkbox[0] = new Checkbox();

// 設定Checkbox類別之文字標籤

checkbox[0].setLabel("Checkbox 0");

// 設定Checkbox類別的選取狀態

checkbox[0].setState(true);

checkbox[0].addItemListener(this);

panel1.add(checkbox[0]);

// 建構函式 2

checkbox[1] = new Checkbox("Checkbox 1");

// 設定Checkbox類別的選取狀態

checkbox[1].setState(false);

checkbox[1].addItemListener(this);

panel1.add(checkbox[1]);

// 建構函式 3

checkbox[2] = new Checkbox("Checkbox 2", true);

checkbox[2].addItemListener(this);

panel1.add(checkbox[2]);

add(panel1);

java.awt.Panel panel2 = new Panel();

// 建構群組

java.awt.CheckboxGroup group = new CheckboxGroup() ;

// 建構函式 1

checkbox[3] = new Checkbox();

// 設定Checkbox類別之文字標籤

checkbox[3].setLabel("Checkbox 3");

// 設定Checkbox類別的群組

checkbox[3].setCheckboxGroup(group);

checkbox[3].addItemListener(this);

panel2.add(checkbox[3]);

// 建構函式 4

checkbox[4] = new Checkbox("Checkbox 4", false, group);

// 設定是否回應互動

checkbox[4].setEnabled(false);

// 設定Checkbox類別的群組

checkbox[4].setCheckboxGroup(group);

checkbox[4].addItemListener(this);

panel2.add(checkbox[4]);

// 建構函式 5

// 設定Checkbox類別的選取狀態

checkbox[5] = new Checkbox("Checkbox 5", group, false);

// 設定Checkbox類別的群組

checkbox[5].setCheckboxGroup(group);

checkbox[5].addItemListener(this);

panel2.add(checkbox[5]);

add(panel2);

// 以CheckboxGroup的setSelectedCheckbox方法

// 設定Checkbox群組中被選取的項目（Item）

group.setSelectedCheckbox(checkbox[3]);

// 設定視窗的大小

this.setSize(400, 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);

}

});

}

public void itemStateChanged(ItemEvent e) {

String source = null;

// 取得產生項目事件的物件

Checkbox cb = (Checkbox)(e.getItemSelectable());

for (int i=0; i < checkbox.length; i++) {

if(cb == checkbox[i]) {

checkbox[i].setState(true);

source = "Select " + checkbox[i].getLabel() + " checkbox";

}

else

checkbox[i].setState(false);

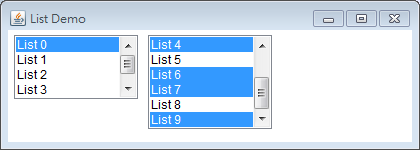
}

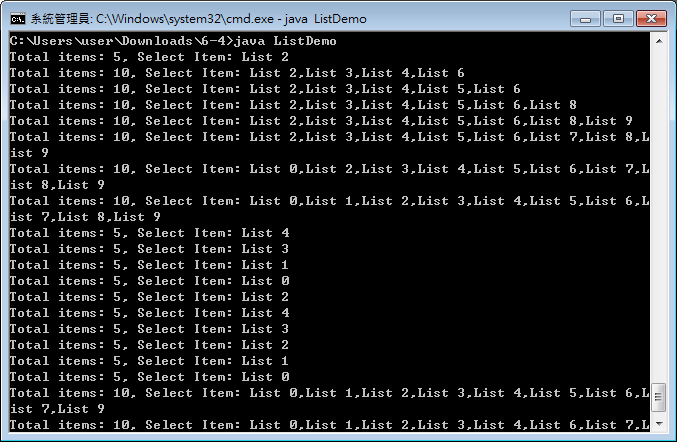
System.out.println(source);

}

}

1. 建立各種樣式的List
   1. 建立List，具有5個項目，不允許複選，預設選取項目為第3項
   2. 建立List，具有10個項目，可顯示列數為6，允許複選，預設選取項目為第2,4,6項
   3. 並請針對上述2個List加上Item Event Listener，點選List產生事件動作並列印出該List的總項目數與被選取到哪些項目了





import java.awt.\*;

import java.awt.event.\*;

public class ListDemo extends java.awt.Frame implements ItemListener {

java.awt.List list;

java.awt.Panel panel;

public static void main(String args[]){

new ListDemo();

}

// 建構函式

public ListDemo() {

super("List Demo");

final int row = 1; // 列

final int column = 3; // 行

// 定義 Layout Manager 為 GridLayout

setLayout(new GridLayout(row, column));

// 建構函式 1

// 預設顯示項目列數為4

// 且不允許多重選擇項目

list = new List();

list.addItemListener(this);

// 設定List選取項目

for (int i=0; i < 5; i++) {

list.add("List " + i);

}

// 設定是否允許項目被多重選擇

list.setMultipleMode(false);

// 選取選項項目

list.select(3);

panel = new Panel();

panel.add(list);

add(panel);

// 建構函式 2

// 可顯示的項目列數為5

list = new List(6);

list.addItemListener(this);

// 設定List選取項目（Item）

for (int i=0; i < 10; i++) {

list.add("List " + i);

}

// 設定是否允許項目被多重選擇

list.setMultipleMode(true);

// 選取選項項目

list.select(2);

list.select(4);

list.select(6);

panel = new Panel();

panel.add(list);

add(panel);

panel = new Panel();

// Demo Only

//panel.setText("Total items: " + list.getItemCount() + ", First Item: " + list.getItem() + ", Init Selected Index: " + list.getSelectedIndex());

add(panel);

// 設定視窗的大小

this.setSize(420, 150);

// 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 itemStateChanged(ItemEvent e) {

String source = null;

// 取得產生項目事件的物件

List li = (List)(e.getItemSelectable());

if(li.isMultipleMode()==true){

source = "Total items: " +li.getItemCount() + ", Select Item: ";

int []iA = li.getSelectedIndexes();

String[]sA=li.getSelectedItems();

System.out.print(source);

for (int i=0; i < iA.length; i++) {

if(i+1 < iA.length)

System.out.print(sA[i]+",");

else

System.out.print(sA[i]);

}

System.out.println("");

}

else{

source = "Total items: " +li.getItemCount() + ", Select Item: " + li.getSelectedItem();

System.out.println(source);

}

}

}

1. 請參考ID\_UI.class，並撰寫具有良好的使用者互動介面的身份證字號產生程式

執行方式為在cmd.exe視窗下鍵入 java ID\_UI

