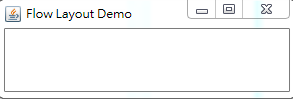
Layout

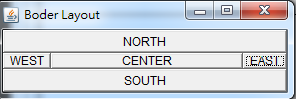
1. 請參考投影片內容，建立以下視窗應用程式

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

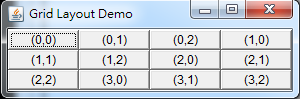


|  |
| --- |
| import java.awt.\*;  import java.awt.event.\*;  public class test1 extends java.awt.Frame {  public static void main(String args[]){  new test1();  }    // 建構函式  public test1() {  super("Flow Layout Demo");  // 向水平中央對齊  FlowLayout flowlayout = new FlowLayout(FlowLayout.CENTER);  // 設定視窗的大小  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);  }  });  }  } |

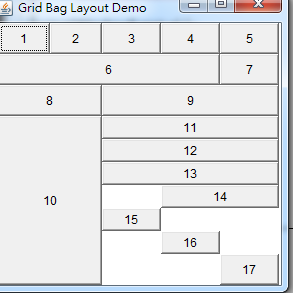
1. 使用Boder Layout建立一基本的視窗程式，結果如下圖

|  |
| --- |
| import java.awt.\*;  import java.awt.event.\*;  public class test2 extends java.awt.Frame {  public static void main(String args[]){  new test2();  }    // 建構函式  public test2() {  super("Boder Layout");    Button button1 = new Button("EAST");  Button button2 = new Button("SOUTH");  Button button3 = new Button("WEST");  Button button4 = new Button("NORTH");  Button button5 = new Button("CENTER");    // 將物件加至Frame中  add(button1,BorderLayout.EAST);  add(button2,BorderLayout.SOUTH);  add(button3,BorderLayout.WEST);  add(button4,BorderLayout.NORTH);  add(button5,BorderLayout.CENTER);  // 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 test3 extends java.awt.Frame {  public static void main(String args[]){  new test3();  }    // 建構函式  public test3() {  super("Grid Layout Demo");    final int row =3;  final int column =4;  setLayout(new GridLayout(row,column));    Button button1 = new Button("(0,0)");  Button button2 = new Button("(0,1)");  Button button3 = new Button("(0,2)");  Button button4 = new Button("(1,0)");  Button button5 = new Button("(1,1)");  Button button6 = new Button("(1,2)");  Button button7 = new Button("(2,0)");  Button button8 = new Button("(2,1)");  Button button9 = new Button("(2,2)");  Button button10 = new Button("(3,0)");  Button button11 = new Button("(3,1)");  Button button12 = new Button("(3,2)");  add(button1);  add(button2);  add(button3);  add(button4);  add(button5);  add(button6);  add(button7);  add(button8);  add(button9);  add(button10);  add(button11);  add(button12);  // 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 test4 extends java.awt.Frame {  public static void main(String args[]){  new test4();  }    // 建構函式  public test4() {  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;  gbConstraints.weightx = 1;  gbConstraints.weighty = 1;      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);    button = new Button("3");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);  button = new Button("4");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  button = new Button("5");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.gridwidth = GridBagConstraints.RELATIVE;  button = new Button("6");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  button = new Button("7");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    // 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 = 1;  // 依加權比例分配物件間垂直方向額外之區域  button = new Button("8");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  button = new Button("9");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    // 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.weighty = 0.0;  gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  gbConstraints.gridheight = 1;  button = new Button("11");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.weighty = 0.0;  gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  gbConstraints.gridheight = 1;  button = new Button("12");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);      gbConstraints.weighty = 0.0;  gbConstraints.gridwidth = GridBagConstraints.REMAINDER;  gbConstraints.gridheight = 1;  button = new Button("13");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);      gbConstraints.gridx=3;  gbConstraints.gridy=6;  button = new Button("14");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);    gbConstraints.gridwidth = 1;  gbConstraints.gridheight = 1;  gbConstraints.gridx=2;  gbConstraints.gridy=7;  button = new Button("15");  gridbaglayout.setConstraints(button, gbConstraints);  add(button);      gbConstraints.gridwidth = 1;  gbConstraints.gridheight = 1;  gbConstraints.gridx=3;  gbConstraints.gridy=8;  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);  // 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);  }  });  }  } |