第二部分
图形界面设计

12 实验 10：JAVA 图形界面的组件与布局

12.1 实验目的

- 了解 Java 系统图形用户界面的工作原理和界面设计步骤。
- 掌握图形用户界面的各种常用组件的使用方法。
- 掌握图形用户界面各种布局策略的设计与使用。

12.2 实验要求

编写一个 Java 应用程序，实现以下要求：显示一个 100*100 的窗口，窗口内添加了四个按钮，其布局为流式布局管理器。当窗口 f 的尺寸被重置后，其 FlowLayout 型的布局也会随之发生变化，各按钮的大小不变，但其相对位置会变化。

12.3 程序模版

请按模板要求，将【代码】替换为 java 程序代码。

TestFlowLayout.java

```
import javax.swing.*;
import java.awt.*;
public class TestFlowLayout {
    public static void main(String args[]) {
        JFrame f = new JFrame("Flow Layout");
        JButton button1 = new JButton("确定");
        JButton button2 = new JButton("打开");
        JButton button3 = new JButton("关闭");
        JButton button4 = new JButton("取消");
        f.setLayout(new FlowLayout());
        【代码 1】 // 在窗口中添加 button1
        【代码 2】 // 在窗口中添加 button2
        【代码 3】 // 在窗口中添加 button3
        【代码 4】 // 在窗口中添加 button4
        f.setSize(100,100);
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

12.4 实验指导

使用 add()方法可以在窗口中添加组建。例如：

```
f.add(button1);
```

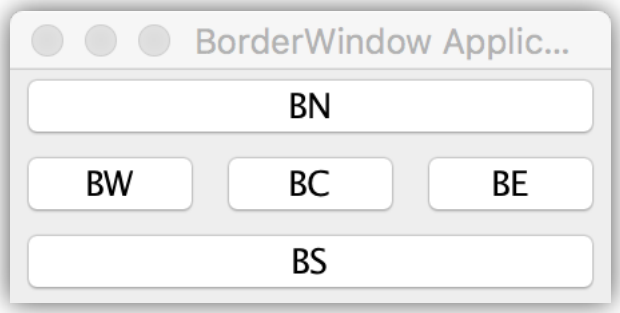
12.5 扩展练习

12.5.1 常用布局

当把组件添加到容器中时，希望控制组件在容器中的位置，这就需要布局设计。上面的实验中使用了 FlowLayout 布局，其他常见的布局还有 BorderLayout，CardLayout，GridLayout 等，修改上述实验代码，尝试其他布局方式。

12.5.2 BORDERLAYOUT

BorderLayout 布局将容器空间简单地划分为东、西、南、北、中 5 个区域，每加入一个组件都应该指明把这个组件加在哪个区域中。区域由 BorderLayout 中的常量 NORTH，SOUTH，EAST，WEST，CENTER 表示。编写一个 Java 应用程序，实现以下窗口布局。

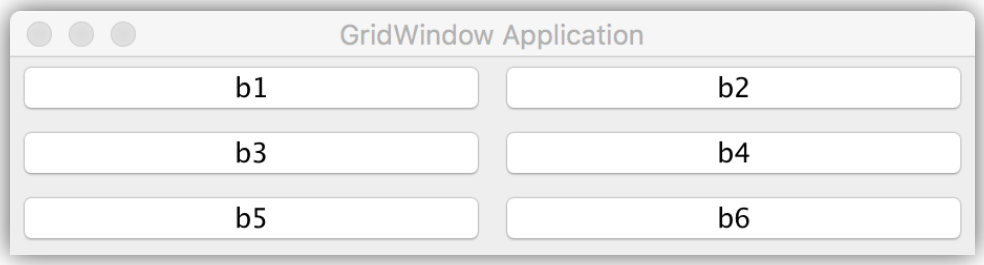


BorderLayoutWindow.java

```
import javax.swing.*;
import java.awt.*;
public class BorderLayoutWindow extends JFrame {
    public BorderLayoutWindow() {
        setLayout(new BorderLayout());
        add(new JButton("BN"),BorderLayout.NORTH);
        add(new JButton("BS"),BorderLayout.SOUTH);
        add(new JButton("BE"),BorderLayout.EAST);
        add(new JButton("BW"),BorderLayout.WEST);
        add(new JButton("BC"),BorderLayout.CENTER);
    }
    public static void main(String args[]) {
        BorderLayoutWindow window = new BorderLayoutWindow();
        window.setTitle("BorderWindow Application");
        window.pack();
        window.setVisible(true);
        window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

12.5.3 GridLayout

GridLayout 将容器划分为若干行*若干列的网格区域，组件就放置在这些划分出来的单元格中。使用 GridLayout 的构造方法 GridLayout(int m, int n)创建布局对象，可以制定划分网格的行数 m 和列数 n。编写一个 Java 应用程序，实现以下窗口布局。



GridLayoutWindow.java

```
import javax.swing.*;
import java.awt.*;
public class GridLayoutWindow extends JFrame {
    public GridLayoutWindow() {
        setLayout(new GridLayout(3,2));
        add(new JButton("b1"));
        add(new JButton("b2"));
        add(new JButton("b3"));
        add(new JButton("b4"));
        add(new JButton("b5"));
        add(new JButton("b6"));
    }
    public static void main(String args[]) {
        GridLayoutWindow window = new GridLayoutWindow();
        window.setTitle("GridWindow Application");
        window.pack();
        window.setVisible(true);
        window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

13 实验 11：事件处理

13.1 实验目的

- 了解 Java 图形用户界面的事件响应机制。
- 掌握鼠标事件编程方法。
- 掌握 AWT 中 Color 和 Font 类的使用方法。

13.2 实验要求

编写一个 Java 应用程序，实现以下要求：创建一个 GridLayout 的框架 f，其标题为 Test。在框架中添加了一个标签为"-"的按钮 b1，一个标签，以及一个标签为"+"的按钮 b2。标签中显示一个数值，初始化为 0。为按钮 b1 和 b2 注册监听器 bh，监听 ActionEvent 事件，当单击框架中的按钮时，会触发 ActionEvent 事件，执行事件处理器 actionPerformed(ActionEvent e)。当点击 b1 按钮则标签中的数值减 1（减到 0 为止），当点击 b2 按钮则标签中的数值加 1。

13.3 程序模版

请按模板要求，将【代码】替换为 java 程序代码。

TestActionEvent.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class TestActionEvent {
    public static void main(String args[]) {
        JFrame f = new JFrame("Test");
        JButton b1 = new JButton("-");
        JButton b2 = new JButton("+");
        JLabel l = new JLabel("0",JLabel.CENTER);
        Monitor bh = new Monitor();
        f.setLayout(new GridLayout());
        bh.setJLabel(l);
        f.add(b1);
        f.add(l);
        f.add(b2);
        b1.setActionCommand("-");
        b2.setActionCommand("+");
        【代码 1】 // 为按钮 b1注册监听器 bh
        【代码 2】 // 为按钮 b2注册监听器 bh
        f.pack();
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
class Monitor implements ActionListener {
      int count = 0;
    JLabel l;
    public void setJLabel (JLabel l) {
        this.l = l;
    }
    public void actionPerformed(ActionEvent e) {
        String str = e.getActionCommand();
        if(str.equals("-")) {
            if(count>0) count--;
        }
        else {
            count++;
        }
        l.setText(""+count);
    }

}
```

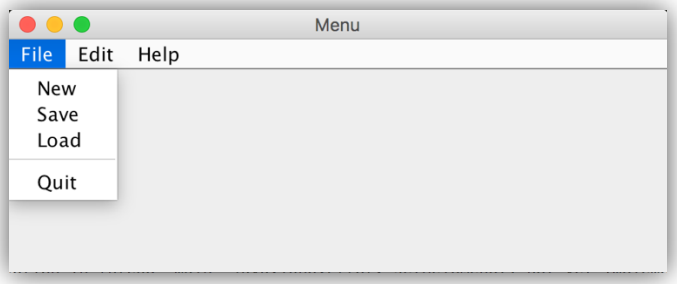
13.4 实验指导

为组件注册监听器可以使用以下语句：

```
addActionListener(监听器);
```

13.5 扩展练习

编写一个 Java 应用程序，实现菜单栏，程序运行的运行结果如下所示：



MenuTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class MenuTest{
    public static void main(String args[]){
        JFrame f = new JFrame("Menu");
        JMenuBar mb = new JMenuBar();
        f.setJMenuBar(mb);
        JMenu m1 = new JMenu("File");
        JMenu m2 = new JMenu("Edit");
        JMenu m3 = new JMenu("Help");
        mb.add(m1);
        mb.add(m2);
        mb.add(m3);
        JMenuItem m11 = new JMenuItem("New");
        JMenuItem m12 = new JMenuItem("Save");
        JMenuItem m13 = new JMenuItem("Load");
        JMenuItem m14 = new JMenuItem("Quit");
        m1.add(m11);
        m1.add(m12);
        m1.add(m13);
        m1.addSeparator();
        m1.add(m14);
        f.pack();
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

14 实验 12：算术测试程序

14.1 实验目的

处理事件时，要很好地掌握事件源、监视器、处理事件的接口之间的关系。事件源是能够产生事件的对象，如文本框、按钮、下拉式列表等。事件源通过调用相应的方法将某个对象作为自己的监视器，事件源增加监视的方法 addXXXListener(XXXListener listener) 中的参数是一个接口，listener 可以引用任何实现了该接口的类创建的对象作为事件源的监视器，当事件源发生

事件时，接口 listener 立刻调用被类实现的接口中的某个负责处理事件源发生的事件。本实验目的是掌握处理 ActionEvent 事件。

14.2 实验要求

编写一个算术测试小软件，用来训练小学生的算术能力。程序由 3 个类组成，其中 Teacher 对象充当监视 器，负责给出算术题目，并判断回答者的答案是否正确。ComputerFrame 对象负责为算术题目提供视图，例如用户可以通过 ComputerFrame 对象提供的 GUI 界面看到题目，并通过该 GUI 界面给出题目的答案；MailClass 是软件的主类。程序运行效果如下图：

14.3 程序模版

请按模版要求，将【代码】替换为 Java 程序代码。

MainClass.java

```
public class MainClass {
  public static void main(String args[]) {
    ComputerFrame frame;
    frame=new ComputerFrame();
    frame.setTitle("算术测试");
    frame.setBounds(100,100,650,180);
  }
}
```

ComputerFrame.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class ComputerFrame extends JFrame {
  JMenuBar menubar;
  JMenu choiceGrade; //选择级别的菜单
  JMenuItem  grade1,grade2;
  JTextField textOne,textTwo,textResult;
  JButton getProblem,giveAnswer;
  JLabel operatorLabel,message;
  Teacher teacherZhang;
  ComputerFrame() {
    teacherZhang=new Teacher();
    teacherZhang.setMaxInteger(20);
    setLayout(new FlowLayout());
    menubar = new JMenuBar();
    choiceGrade = new JMenu("选择级别");
    grade1 = new JMenuItem("幼儿级别");
    grade2 = new JMenuItem("儿童级别");
```

```
    grade1.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {
                                teacherZhang.setMaxInteger(10);
                              }
                             });
    grade2.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {
                                teacherZhang.setMaxInteger(50);
                              }
                             });
    choiceGrade.add(grade1);
    choiceGrade.add(grade2);
    menubar.add(choiceGrade);
    setJMenuBar(menubar);
    【代码 1】            //创建 textOne,其可见字符长是 5
    textTwo=new JTextField(5);
    textResult=new JTextField(5);
    operatorLabel=new JLabel("+");
    operatorLabel.setFont(new Font("Arial",Font.BOLD,20));
    message=new JLabel("你还没有回答呢");
    getProblem=new JButton("获取题目");
    giveAnswer=new JButton("确认答案");
    add(textOne);
    add(operatorLabel);
    add(textTwo);
    add(new JLabel("="));
    add(textResult);
    add(giveAnswer);
    add(message);
    add(getProblem);
    textResult.requestFocus();
    textOne.setEditable(false);
    textTwo.setEditable(false);
    getProblem.setActionCommand("getProblem");
    textResult.setActionCommand("answer");
    giveAnswer.setActionCommand("answer");
    teacherZhang.setJTextField(textOne,textTwo,textResult);
    teacherZhang.setJLabel(operatorLabel,message);
    【代码 2】//将 teacherZhang注册为 getProblem的 ActionEvent事件监视器
    【代码 3】//将 teacherZhang注册为 giveAnswer的 ActionEvent事件监视器
    【代码 4】//将 teacherZhang注册为 textResult的 ActionEvent事件监视器
    setVisible(true);
    validate();
    setDefaultCloseOperation(DISPOSE_ON_CLOSE);
  }
}
```

Teacher.java

```
import java.util.Random;
import java.awt.event.*;
import javax.swing.*;
public class Teacher implements ActionListener {
  int numberOne,numberTwo;
  String operator="";
  boolean isRight;
  Random random;   //用于给出随机数
  int maxInteger;  //题目中最大的整数
  JTextField textOne,textTwo,textResult;
  JLabel operatorLabel,message;
  Teacher() {
    random = new Random();
```

```
  }
  public void setMaxInteger(int n) {
    maxInteger=n;
  }
  public void actionPerformed(ActionEvent e) {
    String str = e.getActionCommand();
    if(str.equals("getProblem")) {
      numberOne = random.nextInt(maxInteger)+1;//1至maxInteger之间的随机数;
      numberTwo=random.nextInt(maxInteger)+1;
      double d=Math.random(); // 获取(0,1)之间的随机数
      if(d>=0.5)
        operator="+";
      else
        operator="-";
      textOne.setText(""+numberOne);
      textTwo.setText(""+numberTwo);
      operatorLabel.setText(operator);
      message.setText("请回答");
      textResult.setText(null);
    }
    else if(str.equals("answer")) {
      String answer=textResult.getText();
      try{
        int result=Integer.parseInt(answer);
        if(operator.equals("+")){
          if(result==numberOne+numberTwo)
            message.setText("你回答正确");
          else
            message.setText("你回答错误");
        }
        else if(operator.equals("-")){
          if(result==numberOne-numberTwo)
            message.setText("你回答正确");
          else
            message.setText("你回答错误");
        }
      }
      catch(NumberFormatException ex) {
        message.setText("请输入数字字符");
      }
    }
  }
  public void setJTextField(JTextField ... t) {
    textOne=t[0];
    textTwo=t[1];
    textResult=t[2];
  }
  public void setJLabel(JLabel ...label) {
    operatorLabel=label[0];
    message=label[1];
  }
}
```

14.4 实验指导

需要将实验中的三个 java 文件保存在同一文件中，分别编译或只编译主类 MainClass，染后运行主类即可。JButton 对象可触发 ActionEvent 事件，JButton 事件源使用 addActionListener 方法获得监视器，创建监视器的类需实现 ActionListener 接口。

14.5 扩展练习

给上述程序增加测试乘法的功能。

15 实验 13：JAVA 异常类

15.1 实验目的

Java 实验 try-catch 语句来处理异常，将可能出现的异常操作放在 try-catch 语句的 try 部分，一旦 try 部分抛出异常对象，例如调用某个抛出异常的方法抛出了异常对象，那么 try 部分将立刻结束执行，而转向执行相应的 catch 部分。本实验的目的是掌握使用 try-catch 语句 。

15.2 实验要求

车站检查危险品的设备，如果发现危险品会发出警告。编程模拟设备发现危险品。

- 编写一个 Exception 的子类 DangerException。该子类可以创建异常对象，该异常象调用 toShow()方法输出"属于危险品"。
- 编写一个 Machine 类，该类的方法 checkBag(Goods goods)当发现参数 goods 是险品时（即 goods 的 isDanger 属性的值是 true 时）将抛出 DangerException 异常。
- 程序在主类的 main 方法中的 try-catch 语句的 try 部分让 Machine 类的实例 checkBag(Goods goods)方法，一旦发现危险品，就在 try-catch 语句的 catch 部分处理危险品。

15.3 程序模板

请按模板要求，将【代码】替换为 java 程序代码。

check.java

```
class Goods {
  boolean isDanger;
  String name;
  Goods(String name) {
    this.name = name;
  }
  public void setIsDanger(boolean boo) {
    isDanger = boo;
  }
  public boolean isDanger() {
    return isDanger;
  }
  public String getName() {
    return name;
  }
}
class DangerException extends Exception {
  String message;
  public DangerException() {
    message = "危险品!";
  }
  public void toShow() {
    System.out.print(message+" ");
  }
```

```
}
class Machine {
  public void checkBag(Goods goods) throws DangerException {
    if(goods.isDanger()) {
      DangerException  danger=new DangerException() ;
      【代码 1】 //抛出 danger
    }
  }
}
public class Check {
  public static void main(String args[]) {
    Machine machine = new Machine();
    Goods apple = new Goods("苹果");
    apple.setIsDanger(false);
    Goods explosive = new Goods("炸药");
    explosive.setIsDanger(true);
    try {
      machine.checkBag(explosive);
      System.out.println(explosive.getName()+"检查通过");
    }
    catch(DangerException e) {
      【代码 2】 //e调用 toShow()方法
      System.out.println(explosive.getName()+"被禁止!");
    }
    try {
      machine.checkBag(apple);
      System.out.println(apple.getName()+"检查通过");
    }
    catch(DangerException e) {
      e.toShow();
      System.out.println(apple.getName()+"被禁止!");
    }
  }
}
```

15.4 实验指导

throw 是 Java 的关键字，该关键字的作用就是抛出异常，因此代码 1 应该是 throw(danger)。

15.5 扩展练习

是否可以将实验代码里 try-catch 语句的 catch 部分捕获的 DangerException 异常更改为 Exception？

是否可以将实验代码里 try-catch 语句的 catch 部分捕获的 DangerException 异常更改为 java.io.IOException？

16 实验 14：StringTokenizer 类

16.1 实验目的

当分析一个字符串并将字符串分解成可被独立使用的单词时，可以使用 java.util 包中 StringTokenizer 类。当我们想分解出字符串的有用的单词时，可以首先把字符串中不需要的单

词都统一替换为空格或其他字符，例如"*"，然后再使用 StringTokenizer 类，并用"*"或空格做分隔标记分解出需要的单词。本实验的目的是掌握 StringTokenizer 类。

16.2 实验要求

两张购物小票的内容如下。

- "苹果 56.7 圆,香蕉:12 圆,芒果:19.8 圆";
- "酱油 6.7 圆,精盐:0.8 圆,榨菜:9.8 圆";

编写程序分别输出两张购物小票的价格之和。

16.3 程序模板

上机调试模板给出的程序，完成实验后的练习。

E.java

```
import java.util.*;
public class E {
    public static void main(String args[ ]) {
        String s1 = "苹果:56.7圆,香蕉:12圆,芒果:19.8圆";
        String s2 = "酱油:6.7圆,精盐:0.8圆,榨菜:9.8圆";
        ComputePice jisuan = new ComputePice();
        String regex = "[^0123456789.]+"; // 匹配所有非数字字符串
        String s1_number = s1.replaceAll(regex,"*");
        double priceSum = jisuan.compute(s1_number,"*");
        System.out.printf("\"%s\"价格总和:\n%f 圆\n", s1, priceSum);
        String s2_number = s2.replaceAll(regex,"#");
        priceSum = jisuan.compute(s2_number, "#");
        System.out.printf("\"%s\"价格总和:\n%f圆\n", s2, priceSum);
    }
}
class ComputePice {
    double compute(String s, String fenge) {
        StringTokenizer fenxiOne = new StringTokenizer(s, fenge);
        double sum = 0;
        double digitItem = 0;
        while(fenxiOne.hasMoreTokens()) {
            String str = fenxiOne.nextToken();
            digitItem = Double.parseDouble(str);
            sum = sum + digitItem;
        }
        return sum;
    }
}
```

16.4 实验指导

如果准备分解出"酱油:6.7 圆,精盐:0.8 圆,榨菜:9.8 圆"的货品名称，即不要价格和价格单位以及标点符号，那么可以实现使用正则表达式"[0123456789.]+圆"匹配诸如 ddddd.ddd 圆的价格数据。那么对于 String temp = s1.replaceAll(re,"");temp 就是字符串:

"酱油:,精盐:,榨菜:"

那么再经过：

```
temp = temp.replaceAll(":"," ");
temp = temp.replaceAll(","," ");
```

之后，temp 就是字符串:

"酱油　精盐　榨菜"

16.5 扩展练习

编写程序输出"酱油:6.7 圆,精盐:0.8 圆,榨菜:9.8 圆"中的货品名称。

17 实验 15：输入输出流

17.1 实验目的

本实验的目的是掌握字符输入输出流以及缓冲输入输出流用法。

17.2 实验要求

现在有如下格式的成绩单（文本格式）score.txt。

姓名:张三,数学 72分,物理 67分,英语 70分.
姓名:李四,数学 92分,物理 98分,英语 88分.
姓名:周五,数学 68分,物理 80分,英语 77分.

要求按行读取成绩单，并在该行的后面加上该字的总成绩，然后将该行写入到一个名字为 scoreAnalysis.txt 的文件中。

17.3 程序模版

请按模版要求，将【代码】替换为 Java 程序代码。

AnalysisResult.java

```
import java.io.*;
import java.util.*;
public class AnalysisResult {
  public static void main(String args[]) {
    File fRead = new File("score.txt");
    File fWrite = new File("socreAnalysis.txt");
    try{
      Writer out = 【代码 1】//以尾加方式创建指向文件 fWrite的 out流
      BufferedWriter bufferWrite = 【代码 2】//创建指向 out的 bufferWrite流
      Reader in = 【代码 3】//创建指向文件 fRead的 in流
      BufferedReader bufferRead =【代码 4】//创建指向 in的 bufferRead流
      String str = null;
      while((str=bufferRead.readLine())!=null) {
        double totalScore=Fenxi.getTotalScore(str);
        str = str+" 总分:"+totalScore;
        System.out.println(str);
        bufferWrite.write(str);
        bufferWrite.newLine();
      }
      bufferRead.close();
      bufferWrite.close();
```

```
    }
    catch(IOException e) {
      System.out.println(e.toString());
    }
  }
}
```

Fenxi.java

```
import java.util.*;
public class Fenxi {
  public static double getTotalScore(String s) {
    Scanner scanner = new Scanner(s);
    scanner.useDelimiter("[^0123456789.]+");
    double totalScore=0;
    while(scanner.hasNext()){
      try{
        double score = scanner.nextDouble();
        totalScore = totalScore+score;
      }
      catch(InputMismatchException exp){
        String t = scanner.next();
      }
    }
    return totalScore;
  }
}
```

17.4 实验指导

因为要以尾加方式创建指向文件 fWrite 的 out 流，即不刷新文件 scoreAnalysis.txt，因此代码 1 可以是：

```
new FileWriter(fWrite,true);
```

17.5 扩展练习

改进程序，使得能统计出每个学生的平均成绩。

18 独立实验任务 2-1（需要提交实验报告）

18.1 实验目的

熟悉 Java 图形界面的基本设计。

18.2 实验任务

编写 Java 应用程序，实现以下登陆界面：

19 独立实验任务 2-2（需要提交实验报告）

19.1 实验目的

熟悉 Java 界面的菜单使用方法。

19.2 实验任务

编写 Java 应用程序，实现以下界面：