作业

（3）参照例子 15 编写一个体现 MVC 结构的 GUI 程序。首先编写一个封装梯形类，然后再编写一个窗口。要求窗口使用 3 个文本框和一个文本区为梯形对象中的数据提供视图，其中 3 个文本框用来显示和更新梯形对象的上底、下底和高，文本区对象用来显示梯形的面积。窗口中有一个按钮，用户单击该按钮后，程序用 3 个文本框中的数据分别作为梯形对象的上底、下底和高，并将计算出的梯形的面积显示在文本区中。

例子


微课视频

9.5 使用 MVC 结构

模型-视图-控制器（Model-View-Controller），简称为 MVC。MVC 是一种先进的设计结构，是 Trygve Reenskaug 教授于 1978 年最早开发的一个基本结构，其目的是以会话形式提供方便的 GUI 支持。MVC 首先出现在 Smalltalk 编程语言中。

MVC 是一种通过 3 个不同部分构造一个软件或组件的理想办法。

- 模型（model）用于存储数据的对象。
- 视图（view）为模型提供数据显示的对象。
- 控制器（controller）处理用户的交互操作，对于用户的操作做出响应，让模型和视图进行必要的交互，即通过视图修改，获取模型中的数据；当模型中的数据变化时，让视图更新显示。

从面向对象的角度看，MVC 结构可以使程序更具有对象化特性，也更容易维护。在设计程序时，可以将某个对象看作“模型”，然后为“模型”提供恰当的显示组件，即“视图”。为了对用户的操作做出响应，可以选择某个组件做“控制器”，当触发事件时，通过“视图”修改或得到“模型”中维护着的数据，并让“视图”更新显示。

在下面的例子 15 中，首先编写一个封装三角形的类（模型角色），然后再编写一个窗口。要求窗口使用 3 个文本框和一个文本区为三角形对象中的数据提供视图，其中 3 个文本框用来显示和更新三角形对象的三个边的长度；文本区对象用来显示三角形的面积。窗口中有一个按钮（控制器角色），用户单击该按钮后，程序用 3 个文本框中的数据分别作为三角形的三个边的长度，并将计算出的三角形的面积显示在文本区中。程序运行效果如图 9.15 所示。

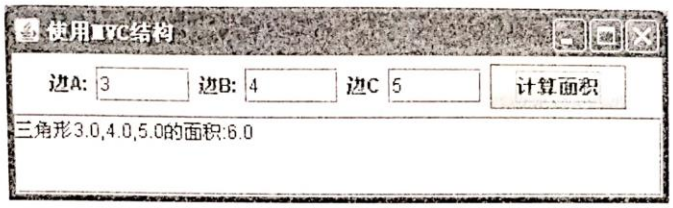

图 9.15　MVC 结构

例子 15

Example9_15.java

```
public class Example9_15 {
   public static void main(String args[]){
      WindowTriangle win = new WindowTriangle();
      win.setTitle("使用 MVC 结构");
      win.setBounds(100,100,420,260);
   }
}
```

WindowTriangle.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class WindowTriangle extends JFrame implements ActionListener {
  Triangle triangle;                    //模型 M
  JTextField textA,textB,textC;         //视图 }V
  JTextArea showArea;                   //视图
  JButton controlButton;                //控制器C
  WindowTriangle() {
    init();
    setVisible(true);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
  void init() {
   triangle = new Triangle();
   textA = new JTextField(5);
   textB = new JTextField(5);
   textC = new JTextField(5);
   showArea = new JTextArea();
   controlButton=new JButton("计算面积");
   JPanel pNorth=new JPanel();
   pNorth.add(new JLabel("边 A:"));
   pNorth.add(textA);
   pNorth.add(new JLabel("边 B:"));
   pNorth.add(textB);
   pNorth.add(new JLabel("边 C"));
   pNorth.add(textC);
   pNorth.add(controlButton);
   controlButton.addActionListener(this);
   add(pNorth,BorderLayout.NORTH);
   add(new JScrollPane(showArea),BorderLayout.CENTER);
  }
  public void actionPerformed(ActionEvent e) {
   try{
      double a = Double.parseDouble(textA.getText().trim());
      double b = Double.parseDouble(textB.getText().trim());
      double c = Double.parseDouble(textC.getText().trim());
      triangle.setA(a);                 //更新数据
      triangle.setB(b);
      triangle.setC(c);
      String area = triangle.getArea();
      showArea.append("三角形"+a+","+b+","+c+"的面积:");
      showArea.append(area+"\n");       //更新视图
   }
   catch(Exception ex) {
      showArea.append("\n"+ex+"\n");
   }
  }
}
```

this 指 win

Triangle.java

```
public class Triangle {
   double sideA,sideB,sideC,area;
   boolean isTriange;
   public String getArea() {
     if(isTriange) {
        double p = (sideA+sideB+sideC)/2.0;
        area = Math.sqrt(p*(p-sideA)*(p-sideB)*(p-sideC));
        return String.valueOf(area);
     }
     else {
        return "无法计算面积";
     }
    }
    public void setA(double a) {
     sideA = a;
     if(sideA+sideB>sideC&&sideA+sideC>sideB&&sideC+sideB>sideA)
       isTriange = true;
     else
       isTriange = false;
    }
    public void setB(double b) {
     sideB = b;
     if(sideA+sideB>sideC&&sideA+sideC>sideB&&sideC+sideB>sideA)
       isTriange = true;
     else
       isTriange = false;
    }
    public void setC(double c) {
     sideC = c;
     if(sideA+sideB>sideC&&sideA+sideC>sideB&&sideC+sideB>sideA)
       isTriange = true;
     else
       isTriange = false;
    }
}
```