**容量计算程序代码**

344121138 陆楠

package Capacity;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.FlowLayout;

import java.awt.GridLayout;

import java.awt.LayoutManager;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JTextField;

public class CapacityCalc\_final extends JFrame implements ActionListener{

// 1. 定义全局变量 全局对象

// 制作一个标签 放置标题

JLabel lblTitle = new JLabel("容量计算系统", JLabel.CENTER);

// 标签数组

JLabel [] lbl = new JLabel[4];

String [] lblStr = {"请输入硬盘容量（GB）", "硬盘的实际容量（GB）", "输入手机内存容量（GB）", "手机的实际容量（GB）"};

// 文本框数组

JTextField [] txt = new JTextField[4];

// 按钮数组

JButton [] btn = new JButton[3];

String [] btnStr = {"计算", "清空", "退出"};

// 面板数组

JPanel [] pnl = new JPanel[4];

Color [] c = {Color.BLUE, Color.pink, Color.green, Color.red};

LayoutManager [] lm = {

new BorderLayout(5, 5), // 东西南北中布局

new FlowLayout(FlowLayout.CENTER, 5, 5), // 自左向右 自上而下

new GridLayout(4, 2, 5, 5), // 4行2列表格

new FlowLayout(FlowLayout.CENTER, 5, 5) // 自左向右 自上而下

};

String[] posStr = {"North", "Center", "South"}; // 三个子面板的方位数组

// 2. 书写构造方法 为所有标签、文本框、按钮初始化

public CapacityCalc\_final() {

// 为4个面板初始化

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

pnl[i] = new JPanel(lm[i]); // 为4个面板上分别设置4个布局

pnl[i].setBackground(c[i]); // 为4个面板上分别设置4个颜色

}

// 北面板上放置标题

pnl[1].add(lblTitle);

// 中面板上放置4个标签及4个文本框

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

lbl[i] = new JLabel(lblStr[i], JLabel.CENTER); // 在4个标签上分别放置4个标题

txt[i] = new JTextField(20); // 初始化4个文本框

}

pnl[2].add(lbl[0]);

pnl[2].add(txt[0]);

pnl[2].add(lbl[1]);

pnl[2].add(txt[1]);

pnl[2].add(lbl[2]);

pnl[2].add(txt[2]);

pnl[2].add(lbl[3]);

pnl[2].add(txt[3]);

// 在南面板上放置3个按钮

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

btn[i] = new JButton(btnStr[i]); // 在三个按钮上分别放置3个标题

btn[i].addActionListener(this); // 为3个按钮添加监听器

pnl[3].add(btn[i]); // 在南面板依次放入按钮

}

// 在主面板上放置3个子面板

for (int i = 0; i < pnl.length - 1; i++) {

pnl[0].add(posStr[i], pnl[i + 1]); // 在主面板上依次放入子面板

}

// 在窗体网格上放置主面板

this.setContentPane(pnl[0]);

}

// 3. 新建整个程序的入口 调用超类方法显示整个窗口内容

public static void main(String [] args) {

CapacityCalc\_final cc = new CapacityCalc\_final();

cc.setTitle("容量计算程序"); // 窗口标题

cc.setBounds(300, 300, 350, 280); // 窗口大小

cc.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

cc.setVisible(true);

}

// 4. 重写超类的抽象方法 实现程序的算法

@Override

public void actionPerformed(ActionEvent ae) {

// TODO Auto-generated method stub

if (ae.getSource() == btn[0]) {

double comInput = Double.parseDouble((txt[0].getText()));

double phoInput = Double.parseDouble((txt[2].getText()));

double comOutput = comInput \* Math.pow(10, 12) / 1024 / 1024 / 1024 / Math.pow(10, 3);

double phoOutput = phoInput \* Math.pow(10, 9) / 1024 / 1024 / Math.pow(10, 3);

txt[1].setText(String.format("%.2f", comOutput));

txt[3].setText(String.format("%.2f", phoOutput));

}

else if (ae.getSource() == btn[1]) {

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

txt[i].setText(""); // 清空4个文本框

}

txt[0].requestFocus(); // 将光标移回第一个文本框

} else System.exit(0); // 退出程序

}

}