# Swing常用组件（一）

## 实验目的

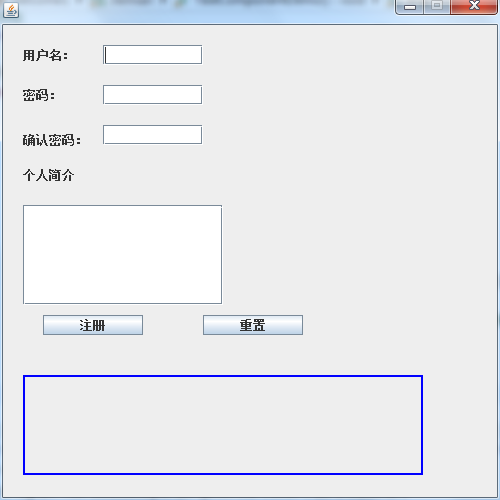
1. 了解Swing的常用组件，了解Swing中常用事件。
2. 可以使用Swing完成简单的界面设计，可以实现简单的人机交互。

## 前提条件

1. 配置完成Eclipse开发环境；
2. 对java中Swing中的常用组件和事件等有一定的了解。
3. 了解Swing中的常用组件JButton、JPanel、JLabel、JTextField、JPasswordField等。

## 实验任务及完成标准

任务一：完成一个如图所示的界面：点击注册按钮时，检测密码和确认密码是否一致，检测用户名是否为空，如果有错误在蓝方块中显示提示信息，如果没有错误，提示注册成功。



注：创建界面的过程为：声明组件--设置组件的位置和属性--将组件添加到指定的容器中—在窗体上显示组件。

步骤：

1. 打开eclipse。
2. 新建一个类文件。
3. 因为是要在窗体中显示所以创建一个JFrame的子类，这样就可以直接从JFrame中继承窗体所有的属性。
4. 在新建的类文件中定义一个类并命名为regedit。在类中声明所需的组件。

**private**JPaneljpContainer = **new**JPanel();//创建一个JPanel对象

**private**JLabeljlUser;//用于显示用户名

**private**JTextFieldjfUser;//用于接收用户输入的用户名

**private**JLabeljlPassword;//用于显示密码

**private**JPasswordFieldjpPassword;//用于接收用户输入的密码

**private**JLabeljlPWDSure;//用于确认显示密码

**private**JPasswordFieldjpPWDSure;//用于接收用户输入的确认密码

**private**JLabeljlRemark;//用于显示备注

**private**JTextAreajaRemark;//用于接收用户输入的个人简介信息

**private**JLabelresult;

**private**JButtonjbInsert1;//注册和重置

**private**JButtonjbInsert2;

1. 声明好所需的组件以后就可以在构造函数中实例化该组件并且设置该组件的属性等。

在构造函数中可以通过以下步骤来使用这么声明的组件

将盛放组件的容器的布局设置为null----实例化组件----定义组件的属性和位置-----为组件添加事件----将组件添加到容器中-----显示组件

1. 将盛放组件的容器的布局设置为null：

jpContainer.setLayout(**null**);

1. 实例化各组件：

//初始化标签对象

jlUser=**new**JLabel("用户名：");

jlPassword=**new**JLabel("密码：");

jlPWDSure=**new**JLabel("确认密码：");

jlRemark=**new**JLabel("个人简介");

//初始化3个文本框对象

jfUser=**new**JTextField();jpPassword=**new**JPasswordField();

jpPWDSure=**new**JPasswordField();

//初始化个人简介文本域对象

jaRemark=**new**JTextArea(4,25); result =**new**JLabel(" ");

//初始化按钮对象

jbInsert1=**new**JButton("注册");

jbInsert2=**new**JButton("重置");

1. 定义组件的属性和位置：

//设置密码框的显示字符为\*

jpPassword.setEchoChar('\*');

jpPWDSure.setEchoChar('\*');

jlUser.setBounds(20,20,70,20);//组件左上角的坐标x,y,宽和高

jlPassword.setBounds(20, 60, 70, 20);

jlPWDSure.setBounds(20, 100, 70, 30);

jfUser.setBounds(100, 20, 100, 20);

jpPassword.setBounds(100, 60, 100, 20);

jpPWDSure.setBounds(100, 100, 100, 20);

jlRemark.setBounds(20, 140, 70, 20)

jaRemark.setLineWrap(**true**);//自动换行

result.setBounds(20,350,400,100);

result.setBorder(BorderFactory.*createLineBorder*(Color.*blue*,2));

jbInsert1.setBounds(40, 290, 100, 20);

jbInsert2.setBounds(200, 290, 100, 20);

//给文本区创建滚动窗体

JScrollPanejsp = **new**JScrollPane(jaRemark);

jsp.setBounds(20, 180, 200, 100);

jpContainer.add(jsp);

1. 为组件添加事件：

//为确定按钮添加事件

jbInsert1.addActionListener(**new**ActionListener() {

@Override

**publicvoid**actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

String strs = "";

**if**(!strs.equals("")){

strs += "\n";

}

String str1=jfUser.getText().toString();

String str2=jpPassword.~~getText~~().toString();

String str3=jpPWDSure.~~getText~~().toString();

**if**(str1.equals("")&&str2.equals("")&&str3.equals("")){

strs+="请将信息填写完整";

}

**elseif**(!str2.equals(str3)) {

strs+="两次密码不一致";

}

**else**{

strs+="注册成功";

}

result.setText(strs);

}

});

//为重置按钮添加事件

jbInsert2.addActionListener(**new**ActionListener() {

**publicvoid**actionPerformed(ActionEvent arg0) {

jfUser.setText("");

jpPassword.setText("");

jpPWDSure.setText("");

jaRemark.setText("");

result.setText("");

}

});

1. 将组件添加到指定的容器：

//将控件添加到面板中的相应位置

jpContainer.add(jlUser);

jpContainer.add(jlPassword);//密码标签

jpContainer.add(jlPWDSure);//确认密码标签

jpContainer.add(jfUser);jpContainer.add(jpPassword);

jpContainer.add(jpPWDSure);jpContainer.add(jlRemark);

jpContainer.add(jaRemark);jpContainer.add(result);

jpContainer.add(jbInsert1);jpContainer.add(jbInsert2);

1. 将容器添加到窗体上并显示组件和窗体。

**this**.setBounds(200,200,500,500);

**this**.setVisible(**true**);

**this**.setResizable(**false**);

**this**.add(jpContainer);

1. 在主函数中生成类的实例。

**publicstaticvoid** main(String[] args){

**new**regedit();

}