第1篇

客户端篇

1. 任务目标

(1) 设计并实现一个完整的聊天室应用程序的客户端界面。

(2) 确保所有界面元素的布局合理、美观，且用户交互流畅。

(3) 实现用户登录、好友列表显示、聊天界面跳转等核心功能。

2. 实现思路

(1) 创建基础窗口：使用 JFrame 类创建应用程序的主窗口。

(2) 布局管理：选择合适的布局管理器(如 BorderLayout、FlowLayout、GridLayout、CardLayout 等)来组织窗口中的控件。

(3) 添加控件：向窗口中添加必要的控件，如 JLabel、JButton、JTextField、JPasswordField、JCheckBox 等。

(4) 设置控件属性：为每个控件设置属性，如文本、大小、位置、可见性等。

(5) 事件监听：为需要交互的控件添加事件监听器(如 ActionListener、MouseListener 等)，以便在用户进行操作时响应。

(6) 卡片布局切换：使用 CardLayout 管理器来实现不同视图之间的切换，如好友列表和陌生人列表。

(7) 数据展示：创建滚动面板来展示数据列表，如好友列表和陌生人列表。

(8) 界面跳转：实现从一个界面到另一个界面的跳转，如从登录界面跳转到好友列表界面，再从好友列表界面跳转到聊天界面。

(9) 聊天功能的实现：创建聊天界面，包括消息显示区域和输入区域，并实现消息的发送和接收。

(10) 用户交互：通过鼠标监听器等实现用户与界面的交互，如双击好友图标打开聊天窗口。

(11) 细节优化：根据需要调整窗口的标题、背景颜色、字体样式等，以提升用户体验。

(12) 程序入口：编写 main 方法作为程序的入口，创建窗口实例并显示。

3. 相关知识点

(1) Swing 控件：了解 Swing 库中各种 GUI 控件的用途和使用方法，如 JFrame、JLabel、JButton、JPanel、JTextField、JPasswordField、JCheckBox 等。

(2) 布局管理：掌握不同的布局管理器，如 BorderLayout、FlowLayout、GridLayout、CardLayout等，以及它们如何控制控件的布局。

(3) 事件处理：理解事件监听和事件处理机制，如何为控件添加事件监听器，以及如何编写事件处理代码。

(4) 数据结构：了解如何使用数据结构(如列表、数组)来存储和管理应用程序中的数据。

(5) 用户界面设计：理解用户界面设计的原则，如何设计直观、易用的用户界面。

(6) 资源管理：学习如何管理和释放应用程序中的资源，如图片、字体等。

4. 模块设计介绍

(1) 主窗口模块(任务 1)：

- 负责创建应用程序的主窗口。
- 设置窗口的基本属性，如大小、标题、关闭操作等。

(2) 登录界面模块(任务 2 和任务 3)：

- 提供用户登录的界面。
- 包含用户名和密码输入框，以及登录按钮。
- 实现登录逻辑，验证用户信息。

(3) 好友列表模块(任务 4～任务 6)：

- 展示用户的好友列表和陌生人列表。
- 使用卡片布局来切换不同的视图。

(4) 聊天界面模块(任务 8)：

- 提供与好友进行聊天的界面。
- 包含消息显示区域和消息输入区域。
- 实现消息的发送和接收逻辑。

(5) 界面跳转模块(任务 7 和任务 9)：

- 负责在不同界面之间进行跳转。
- 包括从登录界面到好友列表界面的跳转，以及从好友列表界面到聊天界面的跳转。

(6) 事件监听模块：

- 负责处理用户的各种交互事件。
- 包括按钮单击事件、列表项选择事件、鼠标单击事件等。

(7) 资源管理模块：

- 负责加载和管理应用程序中的资源，如图片、字体等。
- 确保资源的正确加载和释放。

任务1

第一个简单的窗体


微课视频

CHAPTER 1

1.1 实验任务

完成一个简单的窗体程序，要求创建 JLabel 控件，并设置窗体位置和大小，如图 1-1 所示。

图 1-1

1.2 实现思路

首先创建一个继承自 JFrame 的 ClientLogin 类，用于构建窗体；接着在类中定义一个 JLabel 对象，并设置其文本内容；然后将标签添加到窗体的指定位置（本例中是顶部位置），设置标签的可见性；最后编写 main 方法作为程序入口，创建 ClientLogin 类的实例并显示窗体。

1.3 知识点分析

1.3.1 JFrame 类

Java 提供的 JFrame 类的实例是一个底层容器,即通常所说的窗体,可以向窗体添加控件。使用方法如下。

(1) 创建一个 JFrame 对象:

```
JFrame jFrame = new JFrame("标题");
jFrame.setTitle("标题");
```

(2) 设置窗体的大小和位置:

```
jFrame.setSize(20,10);              //设置了一个长为 20,高为 10 的框图
jFrame.setBounds(1,2,20,10);        //设置一个左上角顶点在(1,2),长为 20,宽为 10 的窗体
jFrame.setLocation(1,2);            //设置一个左上角顶点在(1,2)的窗体
```

(3) 设置窗体可视:

```
jFrame.setVisible(true);
```

1.3.2 JLabel 控件

此控件可以用来显示文本、图像。本身是用来显示信息的,一般情况下是不能直接更改显示器内容的。创建的 JLabel 对象可以通过 Container 类中的 add()方法加入容器中,如图 1-2 所示。

```
1 import javax.swing.JFrame;
2 import javax.swing.JLabel;
3
4 public class SimpleJLabelExample extends JFrame{
5     JLabel jl;
6     public  SimpleJLabelExample() {
7         jl = new JLabel("Java数字聊天室");
8         this.add(jl, "North");
9         this.setBounds(800,600,350,250);
10        this.setVisible(true);
11    }
12    public static void main(String args[]) {
13        SimpleJLabelExample sl = new SimpleJLabelExample();
14    }
15 }
```

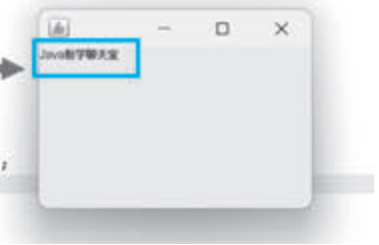

图 1-2

1.3.3 this.setBounds()方法

通过 setBounds(int x, int y, int width, int height)方法对控件进行自定义大小和位置设置。setBounds()有 4 个参数:第 1 个参数为控件在 JFrame 中的 x 坐标;第 2 个参数为控件在 JFrame 中的 y 坐标;第 3 个参数为控件在 JFrame 中的控件宽度;第 4 个参数为控

件在 JFrame 中的控件高度。

1.3.4　this 关键字

在 Java 中，this 是一个引用变量，它引用的是当前对象。在类的方法中，this 可以用来访问当前对象的属性和方法。在本实验中，this 指的是 ClientLogin 类的实例对象，即当前正在创建的窗体对象。

1.3.5　add()方法

add()方法是 Container 类(JFrame 类继承自 Container 类)的一个方法，用于向容器中添加控件。在本实验中，add()方法被用来将 JLabel 控件 jl 添加到 ClientLogin 窗体(也就是 this 指向的 JFrame 对象)中。

1.4　实现代码

```
package com.yychat.view;
import javax.swing.*;
public class ClientLogin extends JFrame{
    JLabel jl;                                          //定义标签控件
    public ClientLogin(){
        jl = new JLabel("Java 教学聊天室");              //创建文本标签控件
        this.add(jl,"North");                           //标签控件添加到窗体顶部

        this.setBounds(800, 600, 350, 250);             //设置窗体边界(位置和大小)
        this.setVisible(true);                          //窗体可视
    }

    public static void main(String[] args) {
        ClientLogin cl = new ClientLogin();             //创建窗体对象
    }
}
```

1.5　实验总结

通过本实验我们可初步了解 Swing 框架中 JFrame 和 JLabel 控件的使用方法，掌握创建简单 GUI 应用程序的基本步骤。在实验中可深刻体会到 GUI 设计对于提升用户体验的重要性，同时也认识到在编程实践中，理论知识与实际操作的结合是不可或缺的。

微课视频

CHAPTER 2

任务2

Java教学聊天室登录窗体的实现1——添加顶部和底部的控件

2.1 实验任务

完成Java教学聊天室的登录和注册窗体程序，要求创建图片JLabel控件、按钮，并设置窗体标题等，如图2-1所示。

图 2-1

2.2 实现思路

在实验中，首先通过Swing库创建一个JFrame窗体，并添加各种Swing控件，如JLabel用于显示文本或图像，JButton用于触发操作；然后用JPanel作为容器，对按钮进行布局管理，采用FlowLayout使其自动调整位置；最后，通过设置窗体的标题、大小和位置等属性，完成整个登录窗体的设计和实现。

2.3　知识点分析

2.3.1　JPanel 类

JPanel 类是 Java Swing 库中的一个类，用于创建面板控件，继承自 Container 类。JPanel 类可以设置背景颜色、边框等属性，并且可以添加其他 Swing 控件（如本实验在面板对象 jp 上添加了 jb1、jb2、jb3 按钮控件）来构造复杂的用户界面。以下是 JPanel 类的相关实验代码：

```
jb1 = new JButton(new ImageIcon("images/login.gif"));        //图片按钮
jb2 = new JButton(new ImageIcon("images/register.gif"));
jb3 = new JButton(new ImageIcon("images/cancel.gif"));
jp = new JPanel();                                            //创建面板对象
jp.add(jb1);                                                  //在 JPanel 面板中添加按钮
jp.add(jb2);                                                  //默认 FlowLayout 流式布局
jp.add(jb3);
this.add(jp,"South");                                         //将 JPanel 控件添加到窗体底部
```

2.3.2　ImageIcon 类

ImageIcon 类是 Java Swing 库中的一个类，用于处理图像图标，继承自 Icon 接口，并提供了一些方法来加载和操作图像。本实验中利用 JButton 的带参构造函数传入 ImageIcon 参数，完成图片按钮 jb1、jb2、jb3 的创建。

```
jb1 = new JButton(new ImageIcon("images/login.gif"));        //图片按钮
jb2 = new JButton(new ImageIcon("images/register.gif"));
jb3 = new JButton(new ImageIcon("images/cancel.gif"));
```

2.3.3　BorderLayout 布局

如图 2-2 所示，容器划分为 5 个区域，每个区域只能放置一个控件，并且随着窗体大小的变化，这些控件会保持其相对位置和大小。如窗体宽度变化时，North/Center/South 进行水平调整，East/West 不发生改变。

图　2-2

2.3.4　setLocationRelativeTo()方法

setLocationRelativeTo()是 Window 类的一个方法,用于设置窗体相对于指定控件的位置。该方法接收一个 Component 类型的参数。如果这个参数是 null,则窗体会被放置在屏幕的中心位置。

图 2-3 和图 2-4 是有无此语句的对比。

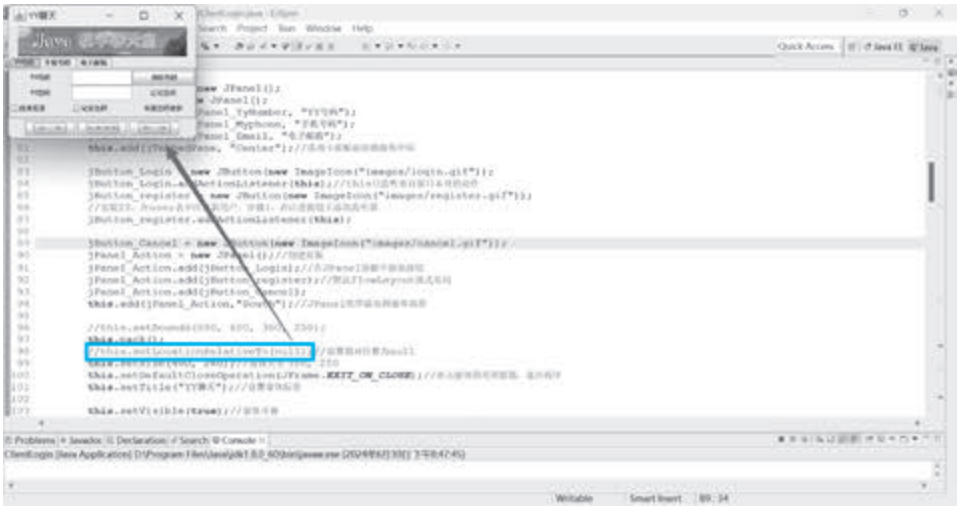

图　2-3

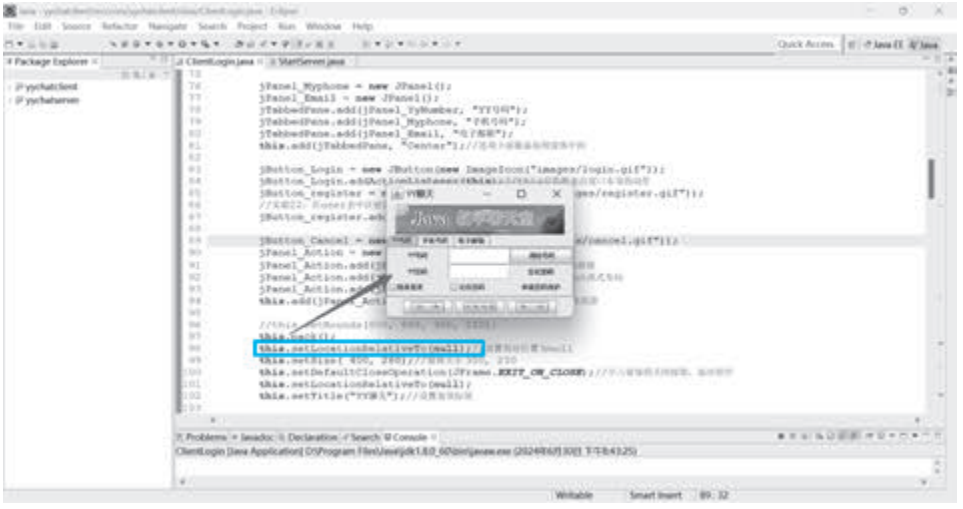

图　2-4

2.3.5　this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)方法

this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)方法可退出应用程序,仅在应用程序中使用。在关闭窗体的同时,终止程序的运行,会带来一定的便利性。默认情况下,该值被设置为 HIDE_ON_CLOSE。如果没有设置的话,默认点关闭时只是隐藏窗体,

在后台进程中还可以看到，如果有多个窗体，只是销毁调用 dispose 的窗体，其他窗体仍然存在，整个应用程序还是处于运行状态。

2.4　实现代码

```
package com.yychat.view;
import javax.swing.*;
public class ClientLogin extends JFrame{
    JLabel jl;                                              //定义标签控件
    JButton jb1,jb2,jb3;
    JPanel jp;
    public ClientLogin(){
        //jl = new JLabel("Java 教学聊天室");                 //文本标签
        jl = new JLabel(new ImageIcon("images/head.gif"));  //图片标签
        this.add(jl,"North");                               //标签控件添加到窗体顶部

        jb1 = new JButton(new ImageIcon("images/login.gif")); //图片按钮
        jb2 = new JButton(new ImageIcon("images/register.gif"));
        jb3 = new JButton(new ImageIcon("images/cancel.gif"));
        jp = new JPanel();                                  //创建面板对象
        jp.add(jb1);                                        //在 JPanel 面板中添加按钮
        jp.add(jb2);                                        //默认 FlowLayout 流式布局
        jp.add(jb3);
        this.add(jp,"South");                               //JPanel 控件添加到窗体底部

        //this.setBounds(800, 600, 350, 250);               //设置窗体边界(位置和大小)
        this.setLocationRelativeTo(null);                   //设置相对位置为 null
        this.setSize(350,250);                              //窗体大小
        this.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE); //单击窗体的关闭按钮,退出程序
        this.setTitle("YY 聊天");                           //设置窗体标题
        this.setVisible(true);                              //窗体可视
    }

    public static void main(String[] args) {
        ClientLogin cl = new ClientLogin();                 //创建窗体对象
    }
}
```

2.5　实验总结

在完成这个实验的过程中，通过实际操作，我们能深入了解图像图标类、面板容器类、流式布局管理器等重要控件和概念。通过使用 Swing 创建用户界面相对简单直观，而且可以通过控件的组合和布局设计出美观实用的界面。在实际操作过程中，我们提升了自己的动手能力和实践能力；在调试过程中，学会了分析问题、查找解决方案并进行调整，这种解决问题的能力对于日后的软件开发工作将会有很大帮助。

任务3

CHAPTER 3

Java教学聊天室登录窗体的实现2——添加中间的选项卡面板

3.1 实验任务

完成Java教学聊天室的登录和注册窗体程序，要求创建图片JLabel控件、按钮，并设置窗体标题等，如图3-1所示。

图 3-1

3.2 实现思路

首先使用Java的Swing库设计一个基础的窗体界面，包括窗体标题、背景颜色、字体样式；然后在窗体上添加标签、文本框、密码框、复选框、按钮等，确保它们的大小和位置符合要求，实现YY号码用户登录界面；最后，设置登录按钮，当用户单击该按钮时，执行登录操作，设置关闭窗体的按钮，当用户单击该按钮时，关闭窗体。

3.3　知识点分析

3.3.1　JTextField 类

JTextField 类是 Java Swing 库中的一个类，用于创建文本输入框。它允许用户输入和编辑单行文本，并且可以设置文本框的大小、是否可见、是否可用等属性。本实验应用于 YY 账号输入框的创立。

```
JTextField jtf = new JTextField();
```

3.3.2　JPasswordField 类

JPasswordField 类是 Java Swing 库中的一个类，用于创建密码输入框。它与 JTextField 类似，但会将用户输入的文本显示为星号或圆点，以保护密码的安全性。

3.3.3　JCheckBox 类

JCheckBox 类是 Java Swing 库中的一个类，用于创建复选框。它允许用户在多个选项中选择一个或多个选项。本实验应用于登录附加选项框“隐身登录”“记住密码”的建立，如图 3-2 所示，其实际功能将在后续实验中实现。

图　3-2

3.3.4　JTabbedPane 控件

JTabbedPane 是一个容器，用于在一个矩形区域中以标签页的形式组织多个控件。每个标签页可以包含不同的控件，如按钮、文本框、图像等。用户可以通过单击标签页切换显示的内容。

3.3.5　GridLayout 布局

GridLayout 布局管理器是一个灵活且功能强大的布局管理工具，用于在界面设计中创建网格状的结构。GridLayout 布局提供的高效且易于管理的布局方式，特别适合于需要规

则网格排列控件的场景,如表单输入、图像画廊或者任何需要网格样式的界面。通过使用 GridLayout 布局管理器,开发者可以创建出既美观又实用的用户界面。

构造方法摘要如下。

(1) GridLayout():创建具有默认值的网格布局。

(2) GridLayout(int rows,int cols):创建具有指定行数和列数的网格布局。

本实验应用第(2)种构造方法创建了 3×3 的网格布局,如图 3-3 所示。

图　3-3

3.4　实现代码

```
package com.yychat.view;
import java.awt.*;
import javax.swing.*;
public class ClientLogin extends JFrame{
    JLabel jl;                                          //定义标签控件
    JButton jb1,jb2,jb3;
    JPanel jp;

    //定义登录界面中间部分的控件
    JLabel jl1,jl2,jl3,jl4;
    JTextField jtf;
    JPasswordField jpf;
    JButton jb4;
    JCheckBox jc1,jc2;
    JPanel jp1,jp2,jp3;
    JTabbedPane jtp;

    public ClientLogin(){
        //jl = new JLabel("Java 教学聊天室");              //文本标签
        jl = new JLabel(new ImageIcon("images/head.gif"));  //图片标签
        this.add(jl,"North");                           //标签控件添加到窗体顶部

        //创建登录界面中间部分的控件
        jl1 = new JLabel("YY 号码:",JLabel.CENTER);
        jl2 = new JLabel("YY 密码:",JLabel.CENTER);
```

```
        jl3 = new JLabel("忘记密码",JLabel.CENTER);
        jl3.setForeground(Color.blue);              //设置字体颜色为蓝色
        jl4 = new JLabel("申请密码保护",JLabel.CENTER);
        jb4 = new JButton(new ImageIcon("images/clear.gif"));
        jtf = new JTextField();
        jpf = new JPasswordField();
        jc1 = new JCheckBox("隐身登录");
        jc2 = new JCheckBox("记住密码");
        jp1 = new JPanel(new GridLayout(3,3));      //设置网格布局模式
        jp1.add(jl1);jp1.add(jtf);jp1.add(jb4);     //在面板中添加 9 个控件
        jp1.add(jl2);jp1.add(jpf);jp1.add(jl3);
        jp1.add(jc1);jp1.add(jc2);jp1.add(jl4);

        jtp = new JTabbedPane();                    //创建选项卡面板
        jtp.add(jp1,"YY 号码");                     //在选项卡中添加 3 个 JPanel 面板
        jp2 = new JPanel();
        jp3 = new JPanel();
        jtp.add(jp2,"手机号码");
        jtp.add(jp3,"电子邮箱");
        this.add(jtp,"Center");                     //选项卡面板添加到窗体中部

        jb1 = new JButton(new ImageIcon("images/login.gif"));//图片按钮
        jb2 = new JButton(new ImageIcon("images/register.gif"));
        jb3 = new JButton(new ImageIcon("images/cancel.gif"));
        jp = new JPanel();                          //创建面板对象
        jp.add(jb1);                                //在 JPanel 面板中添加按钮
        jp.add(jb2);                                //默认 FlowLayout 流式布局
        jp.add(jb3);
        this.add(jp,"South");                       //JPanel 控件添加到窗体底部

        Image im = new ImageIcon("images/duck2.gif").getImage(); //创建 Image 对象
        this.setIconImage(im);                      //设置窗体左上角图标

        //this.setBounds(800, 600, 350, 250);       //设置窗体边界(位置和大小)
        this.setLocationRelativeTo(null);           //设置相对位置为 null
        this.setSize(350,250);                      //窗体大小
        this.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE); //单击窗体的关闭按钮,退出程序
        this.setTitle("YY 聊天");                   //设置窗体标题
        this.setVisible(true);                      //窗体可视
    }
    public static void main(String[] args) {
        ClientLogin cl = new ClientLogin();         //创建窗体对象
    }
}
```

3.5　实验总结

本实验中进一步运用了一系列控件标签(JLabel)、按钮(JButton)、文本框(JTextField)、密码框(JPasswordField)和复选框(JCheckBox)来完成 GUI 的设计,我们可体会到 Java 标准库的便捷实用性。通过本实验,我们对 GUI 框架的设计有了一定的掌握,更加深入地理解了框架、容器、控件之间的相互关系。

任务4

创建好友列表窗体1——实现卡片布局

微课视频

CHAPTER 4

4.1 实验任务

完成 Java 教学聊天室的好友列表窗体的卡片布局，要求创建两张卡片，以及卡片中的三个按钮，如图 4-1 和图 4-2 所示。

图 4-1

图 4-2

4.2 实现思路

首先定义两张卡片面板 JPanel，一张为好友卡片，另一张为陌生人卡片；第一张卡片定义“我的好友”按钮，用 BorderLayout 放到顶部，第二张卡片定义好友按钮和陌生人按钮，放到同一个面板中，使用网格布局放到顶部；最后利用卡片布局管理器 CardLayout 在两张卡片之间切换。

4.3　知识点分析

4.3.1　setLayout()方法

setLayout()是 Container 类的一个方法，用于为容器设置布局管理器，为 Java 图形界面编程的常用方法，用来设置用户界面中屏幕控件的格式布局，默认为流式布局。该方法接收一个实现了 LayoutManager 接口的对象作为参数，常用的有 5 种：FlowLayout、BorderLayout、GridLayout、CardLayout、GridBagLayout。本次实验用到的是 CardLayout。

4.3.2　CardLayout 布局

CardLayout 是 Java Swing 库中的一个布局管理器，用于创建卡片式界面。它允许在同一个容器中切换显示多个面板(Panel)，每个面板可以有自己的布局和控件。在本次实验中，创建一个容器(如 JFrame)作为卡片的容器，并将其布局设置为 CardLayout。创建两个面板，每个面板代表一个卡片的内容。我们可以为每个面板设置不同的布局和控件。将每个面板添加到容器中，并为每个面板指定一个标识符(如"card1""card2"等)。使用 CardLayout 的 show()方法来切换显示不同的卡片，如图 4-3 所示。

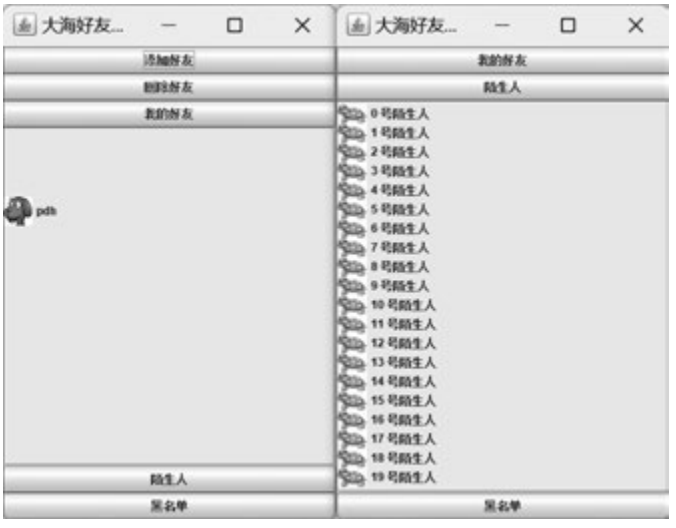

图　4-3

4.3.3　getContentPane()方法

getContentPane()方法是 JFrame 类提供的方法，用于获取内容面板。JFrame 中有一个称为内容面板(Content Pane)的特殊部分，用于容纳窗体中的其他控件，如按钮、标签和文本框等。理解 getContentPane()方法的重要性在于，不能直接向 JFrame 添加控件，而是需要通过其内容面板来管理和布局这些控件。

4.3.4 根窗体(JRootPane)

如图 4-4 所示,根窗体是 Swing 顶级窗体(如 JFrame)的一个核心组成部分,负责管理窗体的边框装饰和可能包含的菜单栏。它位于窗体的最底层,是其他控件的父容器,通常不会直接用来添加控件,而是通过它来管理窗体的整体布局和外观。

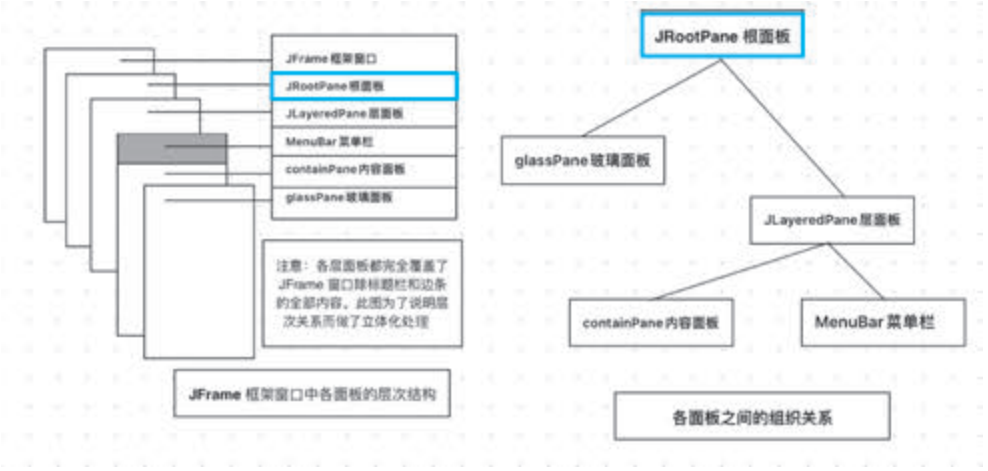

图 4-4

4.3.5 this.setVisible(boolean)方法

this.setVisible(boolean)用来显示/隐藏 GUI 控件,需要显示则使用参数 true,需要隐藏则使用参数 false。setVisible(true)方法的意思是数据模型已经构造好了,允许 JVM 根据数据模型执行 paint 方法开始画图并显示到屏幕上,注意并不是显示图形,而是可以运行开始画图,要把 setVisible()方法放到最后面,代码是按顺序执行的,如果把 setVisible()放在前面,后面再添加其他控件的时候,有可能不会显示出来。

4.4 实现代码

```
package com.yychat.view;

import javax.swing.*;
import java.awt.*;
public class FriendList extends JFrame{

    //定义卡片 1(好友面板)中的控件
    JPanel friendPanel;                                    //卡片 1 中控件的容器
    JButton myFriendButton1;
    JButton myStrangerButton1;
    JButton blackListButton1;
```

```
    //定义卡片 2(陌生人面板)中的控件
    JPanel strangerPanel;                                  //卡片 2 中控件的容器
    JButton myFriendButton2;
    JButton myStrangerButton2;
    JButton blackListButton2;

    public FriendList(){
        //创建卡片 1(好友面板)中的控件
        friendPanel = new JPanel(new BorderLayout());  //卡片 1 设置边界布局模式
        myFriendButton1 = new JButton("我的好友");
        friendPanel.add(myFriendButton1,"North");     //把“我的好友”按钮添加到卡片 1 的顶部

        myStrangerButton1 = new JButton("陌生人");
        blackListButton1 = new JButton("黑名单");
        //为了容纳 myStrangerButton1 和 blackListButton1 定义一个新 JPanel 面板
        JPanel stranger_BlackPanel = new JPanel(new GridLayout(2,1));     //2 行 1 列的网格布局
        stranger_BlackPanel.add(myStrangerButton1);
        stranger_BlackPanel.add(blackListButton1);
        friendPanel.add(stranger_BlackPanel,"South"); //添加到卡片 1 的底部

        //创建卡片 2(陌生人面板)中的控件
        strangerPanel = new JPanel(new BorderLayout());
        myFriendButton2 = new JButton("我的好友");
        myStrangerButton2 = new JButton("陌生人");
        //为了容纳 myFriendButton2 和 myStrangerButton2 定义一个新 JPanel 面板
        JPanel friend_strangerPanel = new JPanel(new GridLayout(2,1)); //2 行 1 列的网格布局
        friend_strangerPanel.add(myFriendButton2);
        friend_strangerPanel.add(myStrangerButton2);
        strangerPanel.add(friend_strangerPanel,"North");              //添加到卡片 2 的顶部

        blackListButton2 = new JButton("黑名单");
        strangerPanel.add(blackListButton2,"South");                  //添加到卡片 2 的底部

        CardLayout cl = new CardLayout();                             //创建卡片布局
        this.setLayout(cl);                                           //窗体设置为卡片布局
        this.add(friendPanel,"card1");                                //窗体中添加卡片
        this.add(strangerPanel,"card2");
        //cl.show(this.getContentPane(), "card1");                    //在窗体中显示卡片 1
        cl.show(this.getContentPane(), "card2");                      //在窗体中显示卡片 2

        this.setIconImage(new ImageIcon("images/duck2.gif").getImage());
        this.setTitle("好友列表");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setBounds(800,600,350,250);
        this.setVisible(true);

    }

    public static void main(String[] args) {
        FriendList fl = new FriendList();
    }
}
```

4.5 实验总结

学习 Java 中创建好友列表并实现卡片布局是一次富有成效的编程实践。在这个过程中,我们首先需要理解卡片布局的基本概念,这是一种在 Java Swing 库中使用的布局管理器,它允许在同一个容器中动态切换不同的视图,同时需要学会如何使用 JPanel 来创建卡片。

任务5

创建好友列表窗体2——实现两张卡片中的列表

CHAPTER 5

微课视频

5.1 实验任务

完成窗体中两张卡片中的列表，要求创建 50 个好友和 20 个陌生人的列表，如图 5-1 和图 5-2 所示。

图 5-1

图 5-2

5.2 实现思路

首先，创建好友列表滚动面板，设置为 50 行 1 列，并通过随机生成图片路径，用 for 循环将 50 个好友标签添加到该面板；随后，创建好友滚动条面板并将面板放在滚动条中；最后，将滚动条面板放到第一张卡片的中部(Center)。陌生人列表滚动面板的实现类似，但数据内容不同。

5.3 知识点分析

5.3.1 滚动条面板

滚动条面板(JScrollPane)是一个容器控件,可以包含其他控件,如文本区域(JTextArea)、列表(JList)、表格(JTable)等。当包含的控件内容超出 JScrollPane 的视野时,它会自动显示出滚动条,如图 5-3 所示,它允许用户通过滚动操作来查看所有内容。JScrollPane 提供了设置水平和垂直滚动条显示策略的功能,例如,可以设置为始终显示滚动条,或者仅在内容超出容器大小时显示。

图 5-3

5.3.2 final 关键字

在 Java 编程语言中,final 是一个关键字,它可以用来修饰类、变量以及成员方法。被 final 修饰的变量,又被称为自定义常量。当一个类、变量或方法被声明为 final 时,它们将不能被继承、修改或重写。这有助于确保类的完整性,提高程序的安全性和可靠性。此外,final 关键字还可以用来实现一些更复杂的编程技术,例如单例模式、枚举类型、可重入锁等,并优化程序性能。

5.3.3 JLabel.LEFT 参数

JLabel.LEFT 参数原型为 static final int LEFT,是用于指定框左侧位置的框方向常量参数。通俗来讲就是文本左对齐,如图 5-4 所示。

5.3.4 对象数组

顾名思义,当数组元素是类对象时,这样的数组称为对象数组,在这种情况下,数组的每一个元素都是一个对象的引用。对象数组的定义与一般数组的定义类似,但是需要将每一个元素实例化。

特别注意:在声明对象数组后,必须对每个数组成员进行实例化之后才能直接使用,否则会报空指针异常!