《Java 程序设计》实验报告 14

学生姓名：李季鸿　　班级：2021 级计本（3）班　　学号：20213002624

实验地点：9 教 304　　指导教师：张春元

实验日期：2023-06-03　　共 4 学时

实验环境：Win10+JDK1.8+ IntelliJ IDEA 2022.1.1

1. 实验目的

学习网络编程和图形图像编程。

2. 实验内容

（1）用集成化开发工具完成实验教材 P132 实验 3 内容。

（2）用集成化开发工具完成实验教材 P147 实验 2 内容。

（3）用集成化开发工具完成实验教材 P149 实验 3 内容。

3. 实验过程

报告撰写具体要求：截屏显示或直接写出实验 1 至实验 3 的源码和运行结果。

实验内容（1）：

Client.java

```
package A.a2;

import java.io.*;
import java.net.*;
import java.util.*;

public class Client {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Socket mysocket = null;
        ObjectInputStream inObject = null;
        ObjectOutputStream outObject = null;
        Thread thread;
        ReadWindow readWindow = null;
        try {
            mysocket = new Socket();
            readWindow = new ReadWindow();
            thread = new Thread(readWindow);
            System.out.print("输入服务器的 IP：");
            String IP = scanner.nextLine();
            System.out.print("输入端口号");
            int port = scanner.nextInt();
            if (mysocket.isConnected()) {
            } else {
                InetAddress address = InetAddress.getByName(IP);
                InetSocketAddress socketAddress = new InetSocketAddress(address, port);
                mysocket.connect(socketAddress);
                InputStream in = mysocket.getInputStream();
                OutputStream out = mysocket.getOutputStream();
                inObject = new ObjectInputStream(in);
                outObject = new ObjectOutputStream(out);
                readWindow.setObjectInputStream(inObject);
                thread.start();
            }
        } catch (Exception e) {
            System.out.println("服务器以断开" + e);
        }
    }
}
class ReadWindow implements Runnable{
    ObjectInputStream in;
    public void setObjectInputStream(ObjectInputStream in){
        this.in=in;
    }
    public void run(){
        double result=0;
        while (true){
            try {
                javax.swing.JFrame window=(javax.swing.JFrame)in.readObject();
                window.setTitle("这是从服务器上读入的窗口");
                window.setVisible(true);
                window.requestFocusInWindow();
                window.setSize(600,800);
```

}

运行结果：

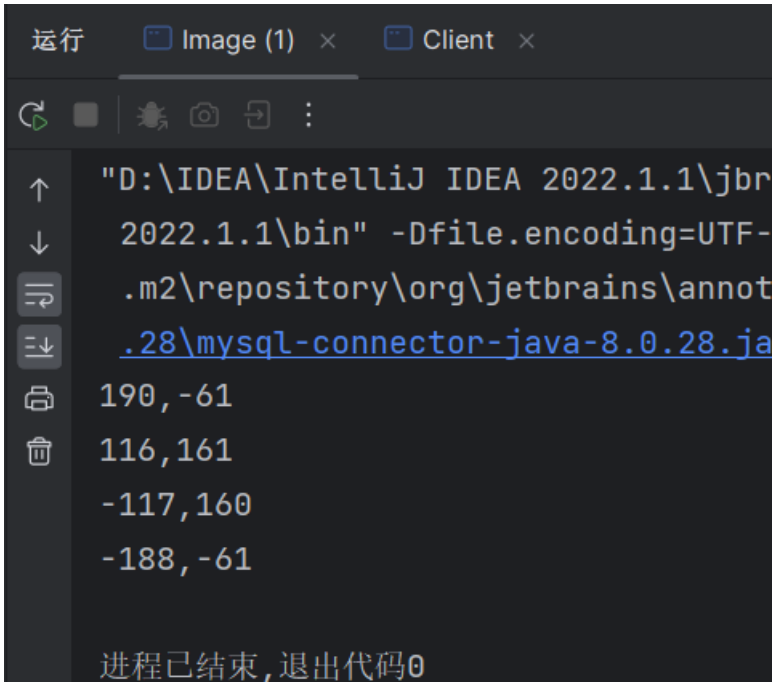
图 14-1　程序运行结果

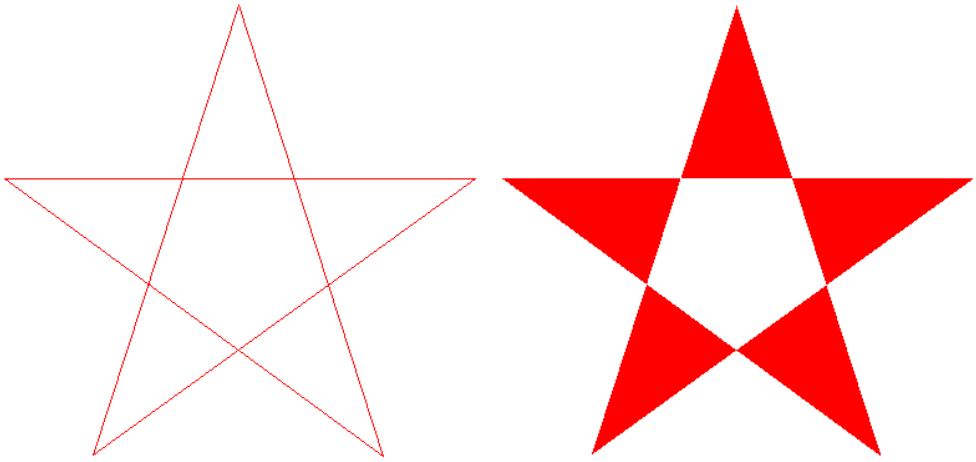
实验内容（2）：

```
package _14_.shiyan2;

import javax.imageio.ImageIO;
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.File;

/**
 * \* Created with IntelliJ IDEA.
 * \* @ProjectName: java_study_codes
 * \* @FileName: Image
 * \* @author: li-jihong
 * \* Date: 2023-05-29 17:37
 */
class PaintCanvas {
    BufferedImage image;
    Graphics2D g_2d;

    PaintCanvas() {
        image = new BufferedImage(1000, 900, BufferedImage.TYPE_INT_RGB);
        g_2d = (Graphics2D) image.createGraphics();
        // 设置背景颜色为白色
        g_2d.setBackground(Color.WHITE);
```

```
            g_2d.clearRect(0, 0, image.getWidth(), image.getHeight());
            int pointX[] = new int[5];
            int pointY[] = new int[5];
            pointX[0] = 0;
            pointY[0] = -200;
            int shiftX = 200;
            int shiftY = 400;
            Polygon polygon1 = new Polygon();
            Polygon polygon2 = new Polygon();
            double arcAngle = (72 * Math.PI) / 180;
            for (int i = 1; i < 5; i++) {
                pointX[i] = (int) (pointX[i - 1] * Math.cos(arcAngle) - pointY[i - 1] *
Math.sin(arcAngle));
                pointY[i] = (int) (pointY[i - 1] * Math.cos(arcAngle) + pointX[i - 1] *
Math.sin(arcAngle));
                System.out.println(pointX[i] + "," + pointY[i]);
            }
            polygon1.addPoint(pointX[0] + shiftX, pointY[0] + shiftY);
            polygon1.addPoint(pointX[2] + shiftX, pointY[2] + shiftY);
            polygon1.addPoint(pointX[4] + shiftX, pointY[4] + shiftY);
            polygon1.addPoint(pointX[1] + shiftX, pointY[1] + shiftY);
            polygon1.addPoint(pointX[3] + shiftX, pointY[3] + shiftY);
            g_2d.setColor(Color.RED);
            g_2d.draw(polygon1);
            polygon2.addPoint(pointX[0] + 3 * shiftX, pointY[0] + shiftY);
            polygon2.addPoint(pointX[2] + 3 * shiftX, pointY[2] + shiftY);
            polygon2.addPoint(pointX[4] + 3 * shiftX, pointY[4] + shiftY);
            polygon2.addPoint(pointX[1] + 3 * shiftX, pointY[1] + shiftY);
            polygon2.addPoint(pointX[3] + 3 * shiftX, pointY[3] + shiftY);
            g_2d.fill(polygon2);
        }

        public BufferedImage getImage() {
            return image;
        }
    }

    public class Image {
        public static void main(String[] args) {
            File file = new File("image/fiveStar.bmp");
            try {
                PaintCanvas draw = new PaintCanvas();
                BufferedImage image = draw.getImage();
```

```
                ImageIO.write(image, "bmp", file);
            } catch (Exception e) {
            }
        }
}
```



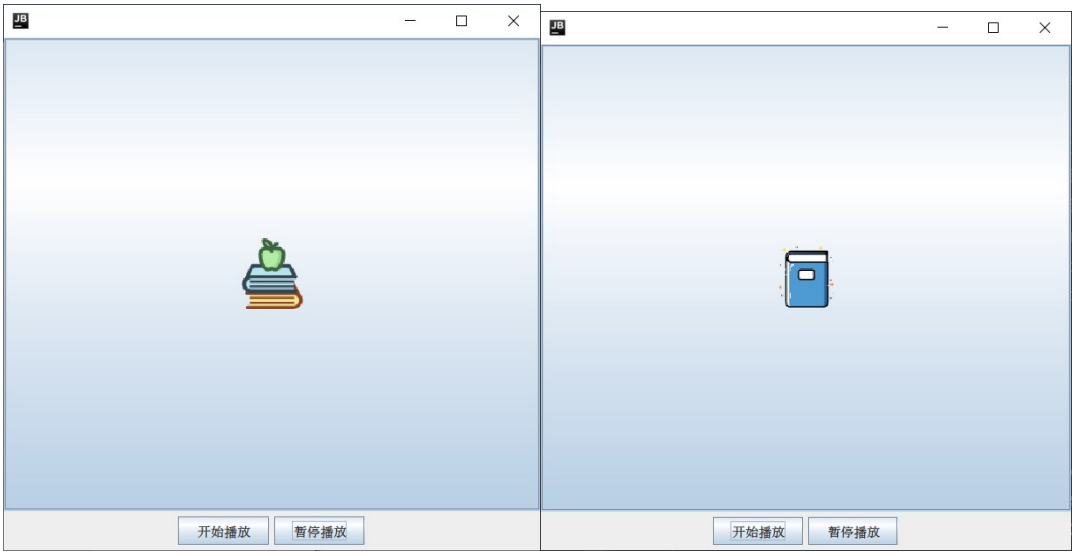


实验内容（3）：

```
package _14_.shiyan3;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener;

/**
 * \* Created with IntelliJ IDEA.
 * \* @ProjectName: java_study_codes
 * \* @FileName: Cartoon
 * \* @author: li-jihong
 * \* Date: 2023-05-29 18:04
 */
class TimeWin extends JFrame implements ActionListener {
    JButton bStart, bStop, imageButton;
    Timer time;
    int n = 0, start = 1, count;
    ImageIcon imageIcon[];

    TimeWin() {
        time = new Timer(500, this);
        imageIcon = new ImageIcon[10];
        count = imageIcon.length;
        for (int i = 0; i < count; i++) {
            imageIcon[i] = new ImageIcon("image/书本  (" + i + ").png");
        }
        imageButton = new JButton(imageIcon[0]);
        bStart = new JButton("开始播放");
        bStop = new JButton("暂停播放");
        bStart.addActionListener(this);
        bStop.addActionListener(this);
        JPanel con = new JPanel();
        con.add(bStart);
        con.add(bStop);
        add(con, BorderLayout.SOUTH);
        add(imageButton, BorderLayout.CENTER);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(500, 500);
        validate();
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == time) {
            n = (n + 1) % count;
            imageButton.setIcon(imageIcon[n]);
        } else if (e.getSource() == bStart) {
```

```
                time.start();
            } else if (e.getSource() == bStop) {
                time.stop();
            }
        }
}

public class Cartoon {
    public static void main(String[] args) {
        TimeWin Win = new TimeWin();
    }
}
```



4. 实验总结

写出实验中的心得体会（对第 13 和 14 章理论课重点简述）。

一、URL 类

URL 类位于 java.net 包中，URL 的实例封装着一个统一资源定位符(Uniform Resource Locator)。

一个 URL 对象通常包含最基本的三部分信息:协议、地址或域名、资源。

使用 URL 创建对象的应用程序称作客户端程序。构造方法

. public URL(String url) throwsMalformedURLException

- public URL(String protocol,String address, Stringfile) throws MalformedURLException

二、InetAdress 类

.用于获取 Internet 或本地主机的域名或 IP 地址静态方法

. InetAdress getByName(String 域名或 IP 地址)用来获得 Internet 上主机的地址对象

. InetAdress getLocalhost( )用来获得本地主机的地址对象实例方法

public string getHostName( )用来获得 InetAddress 对象的域名- public string getHostAddress( )用来获得 InetAddress 对象的 IP 地址

三、套接字

在计算机网络中，我们常采用 IP 地址标识 Internet 上的计算机，用端口号标识正在计算机上运行的进程（程序）。端口号被规定为一个 16 位的 0～65535 之间的整数。

IP 地址与端口号的组合得出一个网络套接字。在 Java 中，当两台主机上的程序需要通信时，可以通过使用 Socket 类和 ServerSocket 类建立客户端套接字对象和服务器端套接字对象并连接在一起。

四、客户端套接字类 Socket

在客户端的进程中使用，负责建立连接到服务器的套接字对象（建立时会抛出 IOException)

构造方法一:

Socket mySocket =

new socket(“服务器 IP 地址”，服务器端口号);注:客户端试图连接服务器端时，后者必须正在运行

构造方法二:

Socket mySocket = new Socket();InetAddress address =

InetAddress.getName(“服务器 IP 地址”)

InetSocketAddress socketAddress =

new InetSocketAddress(address,端口号);mysocket.connect(socketAddress);

五、常用实例方法

getlnputStream()返回一个输入流对象(接收来自套接字对象连接的另一方的输出

流数据)

getOutputStream()返回一个输出流对象(用于向套接字连接的另一方的输入流发送数据)

getlnetAddress()返回一个 InetAddress 对象，在此基础上，可以采用 InetAddress 类提供的方法获得套接字对象所连接的主机名称和地址。

六、服务器端套接字类 ServerSocket

在服务器端的进程中使用,负责建立监听客户端请求的服务器端套接字对象（建立时会抛出 IOException) :ServerSocket serverSocket = new ServerSocket(端口号);

接受客户端连接请求，将服务器端和客户端进程通过套接字对象连接在一起:

Socket socket= serverSocket.accept();与 socket 相关的方法

- getInputStream()返回一个输入流对象(接收来自客户端的输出流数据)

getOutputStream()返回一个输出流对象(用于向客户端的输入流发送数据)

七、Java 图形绘制基本原理

在 Java 中，我们平时看到的一些 Java 组件都是绘制出来，每一个组件类都有一个 public void

paint(Graphics g)方法，程序运行时会将会调用这一个方法完成组件的绘制;用户也可以通过重写这个方法，让程序在组件上绘制出要求的图形。

.自定义图形绘制的一般步骤（例:教材 P447 例 1)

. 1、扩展一个组件类,对其 paint 方法重写如下. 2、在 paint 中采用 Graphics2D 类将参数 g 实例化.3、定义一个基本图形对象

.4、使用 Graphics2D 对象的 fil 或 draw 方法绘制步骤 3 定义的的图形对象

八、仿射变换类 AffineTransform

有时需要平移、缩放或旋转一个图形。可以使用 AffineTransform 类让 Graphics 对象具有变换功能。构造方法

AffineTransform trans=new AffineTransform();实例方法

. translate(double dx,double dy)lI 平移. scale(double sx,double sy)ll 缩放

- rotate(double angle, double x,double y)lI 旋转

注:完成上述图形操作后再将 Graphics 对象设置成具有 trans 功能的画笔,

例:g_2d.setTransform(trans)

九、图形的布尔运算实现步骤

1、两个图形 T1 和 T2 进行布尔运算运算之前，必须分别用这两个图形创建两个 Area 区域对象，例如:. Area a1 = new Area(T1);

- Area a2 =new Area(T2);.2、调用 Area 类的布尔运算方法

- public void intersect(Area r)l 与运算- public void add(Area r)l 或运算

- public void exclusiveOr(Area r)ll 异或运算- public void subtract(Area r)l 差运算

3、然后使用 Graphics2D 对象的 fill 或 draw 方法填充或绘制

十、绘制图像

方法一:利用 Button 类的 setlcon 方法

. lcon icon = new lmage(.lcat.jpg");- button.setlcon(icon) ;

.方法二:

.1、图像加载:首先通过组件调用 getToolkit()方法返回一个 Toolkit 类对象，然后通过该对象调用方法 Image getlmage(String fileNme)或 lmage getImage(File file)即可加载指定的图像。

.2、图像绘制:接下来通过 Graphics2D 对象调用 drawlmage 方法在指定位置绘制。

十一、播放音频的实现步骤

-(1)创建 File 对象

例:File musicFile=new File("hello.wav");(2)获取 URI 对象(URI 类属于 java.net 包)

例:URl uri=musicFile.toURl();

(3)获取 URL 对象

例:URL url=uri.toURL();

-(4)创建音频对象(AudioClip 和 Applet 类属于 java.applet 包)

例:AudioClip clip=Applet.newAudioClip(url);

-(5)播放，循环与停止

例:clip.play()//开始播放

clip.loop()//循环播放，

clip.stop()//停止播放。