南开大学

JAVA语言与应用

客户端/服务器通信程序

实验报告

姓 名：冯朝芃

学 号：2012039

年 级： 2020 级

学 院： 计算机学院

专 业 ：计算机科学与技术

授课教师：刘嘉欣

完成日期：2021年 12月 13日

一、概述：

本作业为客户端/服务器通信程序，本作业实现的功能有：服务端与客户端建立连接，服务端、客户端选择通信端口，服务端、客户端相互发送消息等。

二、运行展示：

运行效果截图：

图形用户界面, 应用程序, Teams

描述已自动生成

附录：完整代码（服务端）

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Manage.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package manager;

import viewer.Viewer;

import javax.swing.\*;

public class Manage {

    private JFrame mainWindow;

    private Viewer viewer;

    void init(){

        mainWindow = new JFrame();

        viewer = new Viewer(mainWindow);

        viewer.init();

        viewer.show();

        while(true){

            //viewer.drawPanel.drawLine(200,200,100,100);

            viewer.update();

        }

    }

    public static void main(String[] args){

        Manage manage = new Manage();

        manage.init();

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『store.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package storer;

import java.util.Queue;

public class store {

    public Queue<String> queue;

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『background.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.\*;

public class background extends JPanel {

    background(){

        setLayout(new GridLayout(1,2));

        setBackground(new java.awt.Color(255, 255, 255));

        setPreferredSize(new java.awt.Dimension(50, 50));

        JLabel label = new JLabel("Background");

        label.setHorizontalAlignment(JLabel.CENTER);

        add(label);

        JComboBox comboBox = new JComboBox();

        comboBox.addItem("Red");

        comboBox.addItem("Green");

        comboBox.addItem("Blue");

        add(comboBox);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『buttonSet.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Vector;

class triangularButton extends JButton {

    triangularButton() {

        super("Triangle");

        setPreferredSize(new java.awt.Dimension(100, 80));

        setContentAreaFilled(true);

        setVisible(true);

        //addActionListener(new Actiona);

    }

}

class rectangularButton extends JButton {

    rectangularButton(){

        super("Rectangle");

        setPreferredSize(new java.awt.Dimension(100, 80));

        setContentAreaFilled(true);

        setVisible(true);

    }

}

class ovalButton extends JButton {

    ovalButton(){

        super("Oval");

        setPreferredSize(new java.awt.Dimension(100, 80));

        setContentAreaFilled(true);

        setVisible(true);

    }

}

class roundButton extends JButton {

    roundButton(){

        super("Round");

        setPreferredSize(new java.awt.Dimension(100, 80));

        setContentAreaFilled(true);

        setVisible(true);

    }

}

class lineButton extends JButton {

    lineButton(){

        super("Line");

        setPreferredSize(new java.awt.Dimension(100, 80));

        setContentAreaFilled(true);

        setVisible(true);

    }

}

public class buttonSet extends JButton {

    public static Vector<JButton> buttons;

    public static JButton selected=new lineButton();

    class buttonListener implements ActionListener {

        @Override

        public void actionPerformed(ActionEvent e) {

            selected = (JButton) e.getSource();

            //System.out.println("Selected: " + selected.getText());

        }

    }

    public buttonSet() {

        buttons = new Vector<JButton>();

        buttons.add(new triangularButton());

        buttons.add(new rectangularButton());

        buttons.add(new ovalButton());

        buttons.add(new roundButton());

        buttons.add(new lineButton());

        for (JButton b : buttons) {

            b.addActionListener(new buttonListener());

        }

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『drawPanel.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.MouseListener;

import java.awt.geom.\*;

import static java.lang.Math.abs;

public class drawPanel extends JPanel {

    public int x,x1,x2;

    public int y,y1,y2;

    private int width = 0;

    private int height = 0;

    private Graphics g;

    // Constructor

    public drawPanel() {

        super();

        setPreferredSize(new java.awt.Dimension(500, 500));

        setBackground(java.awt.Color.white);

        //setVisible(true);

        addMouseListener(new MouseListener() {

            @Override

            public void mouseClicked(java.awt.event.MouseEvent e) {

                x = e.getX();

                y = e.getY();

                //repaint();

            }

            @Override

            public void mousePressed(java.awt.event.MouseEvent e) {

                x1 = e.getX();

                y1 = e.getY();

                repaint();

            }

            @Override

            public void mouseReleased(java.awt.event.MouseEvent e) {

                x2 = e.getX();

                y2 = e.getY();

                repaint();

                //x1 = 0;

                //x2 = 0;

                //y1 = 0;

                //y2 = 0;

            }

            @Override

            public void mouseEntered(java.awt.event.MouseEvent e) {

//                x1 = e.getX();

//                y1 = e.getY();

//                x2 = e.getX();

//                y2 = e.getY();

                repaint();

            }

            @Override

            public void mouseExited(java.awt.event.MouseEvent e) {

                x = e.getX();

                y = e.getY();

                //repaint();

            }

        }

        );

    }

    @Override

    public void paint(Graphics g) {

        super.paint(g);

        //g.drawLine(200, 200, 400, 400);

    }

    @Override

    public void repaint() {

        super.repaint();

        //drawLine(x1, y1, x2, y2);

    }

    //draw a line on the panel with the given coordinates

    public void drawLine(int x1, int y1, int x2, int y2) {

        Graphics g = getGraphics();

        g.drawLine(x1, y1, x2, y2);

    }

    public void drawRectangle(int x1, int y1, int x2, int y2,boolean fill, Color color) {

        Graphics g = getGraphics();

        if(fill) {

            g.setColor(color);

            g.fillRect(x1, y1, abs(x1-x2), abs(y1-y2));

        }else {

            g.drawRect(x1, y1, abs(x1 - x2), abs(y1 - y2));

        }

    }

    public void drawCircle(int x1, int y1, int x2, int y2,boolean fill, Color color) {

        Graphics g = getGraphics();

        if(fill){

            g.setColor(color);

            g.fillOval(x1, y1, abs(x1-x2), abs(y1-y2));

        }else {

            g.drawOval(x1, y1, abs(x1-x2), abs(y1-y2));

        }

    }

    public void drawTriangle(int x1, int y1, int x2, int y2,boolean fill, Color color) {

        Graphics g = getGraphics();

        int[] xPoints = {x1, x2, (x1+x2)/2};

        int[] yPoints = {y1, y2, (y1+y2)/2};

        if(fill){

            g.setColor(color);

            g.fillPolygon(xPoints, yPoints, 3);

        }else

        g.drawPolygon(xPoints, yPoints, 3);

    }

    public void drawRound(int x1, int y1, int x2, int y2,boolean fill, Color color) {

        Graphics g = getGraphics();

        if(fill) {

            g.setColor(color);

            g.fillRoundRect(x1, y1, abs(x1-x2), abs(y1-y2), abs(x1-x2), abs(y1-y2));

        }else

        g.drawRoundRect(x1, y1, abs(x1-x2), abs(y1-y2), 10, 10);

    }

    public void drawText(int x, int y,String text, Color color) {

        Graphics g = getGraphics();

        g.setColor(color);

        g.drawString(text, x, y);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『fillRegion.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import javax.swing.event.ChangeListener;

import java.awt.\*;

public class fillRegion extends JPanel {

    public JCheckBox checkBox;

    public boolean isChecked;

    class fillRegionListener implements ChangeListener {

        public void stateChanged(javax.swing.event.ChangeEvent e) {

            isChecked = checkBox.isSelected();

        }

    }

    fillRegion(){

        setLayout(new GridLayout(1,1));

        setVisible(true);

        checkBox = new JCheckBox("Fill Region");

        checkBox.addChangeListener(new fillRegionListener());

        add(checkBox);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『leftControlBar.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.util.Vector;

public class leftControlBar extends JPanel {

    public static buttonSet buttonSet;

    public void addAll(Vector buttons){

        for(Object button : buttons){

            this.add((JComponent)button);

        }

    }

    leftControlBar(){

        this.setPreferredSize(new java.awt.Dimension(100, 550));

        this.setBackground(new java.awt.Color(108, 108, 108));

        //this.setVerticalScrollBarPolicy(JScrollPane.VERTICAL\_SCROLLBAR\_NEVER);

        //this.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL\_SCROLLBAR\_AS\_NEEDED);

        buttonSet = new buttonSet();

        this.addAll(viewer.buttonSet.buttons);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『shapeColor.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import javax.swing.event.ChangeListener;

import java.awt.\*;

import java.awt.event.ActionListener;

public class shapeColor extends JPanel {

    public Color color=Color.RED;

    public JComboBox comboBox;

    class ComboBoxListener implements ActionListener {

        public void actionPerformed(java.awt.event.ActionEvent e) {

            //convert to Color

            String colorName = (String) comboBox.getSelectedItem();

            color = Color.decode(colorName);

        }

    }

    shapeColor(){

        setLayout(new GridLayout(1,2));

        setBackground(new java.awt.Color(255, 255, 255));

        setPreferredSize(new java.awt.Dimension(50, 50));

        JLabel label = new JLabel("Set Color");

        label.setHorizontalAlignment(JLabel.CENTER);

        add(label);

        comboBox = new JComboBox();

        comboBox.addItem("Red");

        comboBox.addItem("Green");

        comboBox.addItem("Blue");

        comboBox.addActionListener(new ComboBoxListener());

        add(comboBox);

    }

    public Color getColor() {

        return color;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『textSetter.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.\*;

public class textSetter extends JPanel {

    textSetter(){

        setLayout(new GridLayout(1,3));

        JTextField textField = new JTextField();

        textField.setText("Enter text here");

        add(textField);

        JLabel label = new JLabel("size:");

        add(label);

        JTextField fontSize = new JTextField("14");

        fontSize.add(new JScrollBar(JScrollBar.HORIZONTAL));

        add(fontSize);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『TopPanel.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.\*;

public class TopPanel extends JPanel {

    public static shapeColor color;

    public static background background;

    public static textSetter textSetter;

    public static fillRegion fillRegion;

    TopPanel() {

        setLayout(new GridLayout(1,2));

        setPreferredSize(new Dimension(800, 50));

        setBackground(Color.WHITE);

        setBorder(BorderFactory.createMatteBorder(0, 0, 1, 0, Color.BLACK));

        JPanel leftPanel = new JPanel();

        leftPanel.setLayout(new GridLayout(2,1));

        leftPanel.setPreferredSize(new Dimension(400, 23));

        leftPanel.setBackground(Color.WHITE);

        color = new shapeColor();

        leftPanel.add(color);

        background = new background();

        leftPanel.add(background);

        JPanel rightPanel = new JPanel();

        rightPanel.setLayout(new GridLayout(2,1));

        rightPanel.setPreferredSize(new Dimension(400, 23));

        rightPanel.setBackground(Color.PINK);

        textSetter = new textSetter();

        rightPanel.add(textSetter);

        fillRegion = new fillRegion();

        rightPanel.add(fillRegion);

        this.add(leftPanel);

        this.add(rightPanel);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Viewer.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import javax.swing.\*;

import java.awt.\*;

public class Viewer {

    private JFrame mainFrame;

    private TopPanel topPanel;

    private leftControlBar leftControlBar;

    //private buttonSet buttonSet;

    private drawPanel drawPanel;

    public Viewer(JFrame main) {

        mainFrame = main;

    }

    public void update() {

        switch(buttonSet.selected.getText()){

            case "Line":

                System.out.println("draw Line");

                drawPanel.drawLine(drawPanel.x1,drawPanel.y1, drawPanel.x2,drawPanel.y2);

                break;

                case "Rectangle":

                    System.out.println("draw Rectangle");

                drawPanel.drawRectangle(drawPanel.x1,drawPanel.y1, drawPanel.x2,drawPanel.y2,topPanel.fillRegion.isChecked,topPanel.color.getColor());

                break;

                case "Oval":

                drawPanel.drawCircle(drawPanel.x1,drawPanel.y1, drawPanel.x2,drawPanel.y2,topPanel.fillRegion.isChecked,topPanel.color.getColor());

                break;

                case "Triangle":

                drawPanel.drawTriangle(drawPanel.x1,drawPanel.y1, drawPanel.x2,drawPanel.y2,topPanel.fillRegion.isChecked,topPanel.color.getColor());

                break;

            case"Round":

                drawPanel.drawRound(drawPanel.x1,drawPanel.y1, drawPanel.x2,drawPanel.y2,topPanel.fillRegion.isChecked,topPanel.color.getColor());

                break;

            default:

        }

        //mainFrame.repaint();

    }

    public void init() {

        mainFrame.setTitle("Painter");

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

        mainFrame.setSize(800, 600);

        mainFrame.setLocationRelativeTo(null);

        mainFrame.setBackground(new java.awt.Color(255, 255, 255));

        mainFrame.setLayout(new BorderLayout());

        topPanel = new TopPanel();

        mainFrame.add(topPanel, BorderLayout.NORTH);

        leftControlBar = new viewer.leftControlBar();

        mainFrame.add(leftControlBar, BorderLayout.WEST);

        drawPanel = new drawPanel();

        mainFrame.add(drawPanel, BorderLayout.CENTER);

    }

    public void show() {

        mainFrame.setVisible(true);

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Beans.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

import java.net.ServerSocket;

public class Beans {

    public Event event;

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Event.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

public enum Event {

    START\_SERVER,SET\_CLIENT,SEND\_MESSAGE,GET\_MESSAGE,NO\_MESSAGE;

    //public Event event;

    public static Event getEvent(String str){

        if(str.equals("START\_SERVER")){

            return START\_SERVER;

        }else if(str.equals("SET\_CLIENT")){

            return SET\_CLIENT;

        }else if(str.equals("SEND\_MESSAGE")){

            return SEND\_MESSAGE;

        }else if(str.equals("GET\_MESSAGE")){

            return GET\_MESSAGE;

        }

        return null;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『noResponse.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

public class noResponse extends Beans{

    private int who;

    public noResponse(Event event,int n){

        this.event = event;

        this.who = n;

    }

    public int getWho() {

        return who;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『sendMessage.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

public class sendMessage extends Beans{

    //private int port;

    private String message;

    public sendMessage(String message) {

        this.message = message;

        this.event = Event.SEND\_MESSAGE;

    }

    public String getMessage() {

        return message;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『startService.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

public class startService extends Beans {

    private int port;

    private int who;

    public startService(int port,int w) {

        this.port=port;

        this.event = Event.START\_SERVER;

        this.who=w;

    }

    public int getPort() {

        return port;

    }

    public int getWho() {

        return who;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『TextResponse.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package event;

public class TextResponse extends Beans{

    private int who;

    private String res;

    public TextResponse(Event event,int n,String s){

        this.event = event;

        this.who = n;

        this.res = s;

    }

    public int getWho() {

        return who;

    }

    public String getRes() {

        return res;

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Manager.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package mainWindow;

import event.Beans;

import event.sendMessage;

import event.startService;

import netUtil.net;

import viewer.Viewer;

import event.Beans.\*;

import java.util.\*;

import java.util.concurrent.\*;

public class Manager {

    private static Manager m;

    private static Viewer viewer;

    private Map<String,Integer> clientThreads;

    private static ExecutorService exe;

    private int port;

    private static net n;

    private void init() {

        viewer= new Viewer();

        viewer.start(this);

        //n=new net(this);

    }

    public static void main(String[] args) {

        m=new Manager();

        m.init();

        exe = Executors.newCachedThreadPool();

    }

    public void sendMessage(Beans bean) {

//        exe.execute(new Thread(){

//           public void run() {

               //net n=new net("localhost",port);

               n.sendData(((sendMessage)bean).getMessage());

//            }

//        });

    }

    public void startService(Beans bean) {

        port=((startService)bean).getPort();

        exe.execute(new Thread(){

            public void run() {

                n=new net("localhost",port,m);

                n.startServe(((startService)bean).getWho());

            }

        });

    }

    public void noResponse(Beans b){

        viewer.noResponse(b);

    }

    public void getMessage(Beans b){

        viewer.getMessage(b);

    }

    public void closing() {

        n.closeServer();

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『net.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package netUtil;

import event.Event;

import event.TextResponse;

import event.noResponse;

import mainWindow.Manager;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.net.InetAddress;

import java.net.ServerSocket;

import java.net.Socket;

public class net {

    public static final int SMALL\_BUF\_SIZE = 144;

    private Manager m;

    private InetAddress ip;

    private volatile Socket socket;

    private int port;

    private String host;

    private volatile ServerSocket ss;

    private boolean isServer;

    private String gotData;

    private String sentData;

    private int clientNumber=1;

//    public net(Manager m){

//        this.m = m;

//    }

    public net(String host, int port,Manager m) {

        this.m = m;

        this.host = host;

        this.port = port;

        //socket = new Socket(host, port);

        try{

            ss=new ServerSocket(port);//,0,InetAddress.getByName(host));

        }catch (Exception e){

            e.printStackTrace();

        }

    }

    public void startServe(int c) {

        clientNumber=c;

        try{

            socket = ss.accept();

            isServer=true;

            this.getMessage();

            //this.wait();

        }catch(Exception e){

            e.printStackTrace();

        }

//        finally{

//            this.closeServer();

//        }

    }

    public void closeServer(){

        try {

            socket.close();

            ss.close();

        }catch(Exception e){

            e.printStackTrace();

        }

        isServer=false;

    }

    public void getData(){

        try{

            gotData=socket.getInputStream().toString();

        }catch(Exception e){

            e.printStackTrace();

        }

    }

    public void sendData(String str){

        sentData=str;

        try{

            socket.getOutputStream().write(sentData.getBytes());

            socket.getOutputStream().flush();

        }catch(Exception e){

            e.printStackTrace();

            m.noResponse(new noResponse(Event.NO\_MESSAGE,clientNumber));

        }

    }

    public void getMessage(){

        Thread t= new Thread(()->{

            while(true) {

                try {

                    BufferedReader br= null;

                    InputStream in=socket.getInputStream();

                    br = new BufferedReader(new InputStreamReader(in,"utf8"));

                    StringBuilder reqStr = new StringBuilder();

                    char[] buf = new char[SMALL\_BUF\_SIZE];

                    do {

//                        String str = br.readLine();//自循环 流没有结束符 不知道哪里是一行 阻塞

//                        m.getMessage(new TextResponse(Event.GET\_MESSAGE, str));

                        if (br.read(buf) != -1) {

                            reqStr.append(buf);

                        }

                    }while(br.ready());

                    String str=reqStr.toString();

                    //socket.shutdownInput();

                    //System.out.println("got: "+in);

                    //System.out.println(socket.isClosed());

                    if(str!=null) {

                        m.getMessage(new TextResponse(Event.GET\_MESSAGE,clientNumber, str));

                    }

                } catch (Exception e) {

                    e.printStackTrace();

                    break;

                }

            }

        });

        //t.setDaemon(true);

        t.start();

    }

}

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

『Viewer.java』

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

package viewer;

import event.\*;

import mainWindow.Manager;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import static java.lang.Integer.parseInt;

public class Viewer extends JFrame {

    private Manager m;

    private JPanel panel;

    private JComboBox portSelectorBox;

    private JComboBox clientSelectorBox;

    private JTextArea textArea;

    private JTextArea logArea;

    private JTextField statusField;

    private JButton sendButton;

    private JButton startButton;

    private StringBuilder lowTextBuffer=new StringBuilder();

    @Override

    protected void processWindowEvent(WindowEvent e) {

        if(e.getID()==WindowEvent.WINDOW\_CLOSING) {

            m.closing();

        }

        super.processWindowEvent(e);

    }

    public void start(Manager manager) {

        m=manager;

        init();

        setVisible(true);

    }

    public void noResponse(Beans b) {

        lowTextBuffer.append("Client"+(((noResponse)b).getWho()+1)+" Sending No Message\n");

        logArea.setText(String.valueOf(lowTextBuffer));

    }

    public void getMessage(Beans b) {

        lowTextBuffer.append("Client"+(((TextResponse)b).getWho()+1)+" "+((TextResponse)b).getRes()+"\n");

        logArea.setText(String.valueOf(lowTextBuffer));

    }

    private void init() {

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setSize(400, 400);

        setLocationRelativeTo(null);

        setTitle("Server");

        panel = new JPanel();

        panel.setLayout(new GridLayout(5,1));

        add(panel);

        panel.add(new JPanel());

        ((JPanel)panel.getComponent(0)).setLayout(new FlowLayout());;

        portSelectorBox = new JComboBox();

        portSelectorBox.addItem("Select port");

        portSelectorBox.addItem("8080");

        portSelectorBox.addItem("10000");

        ((JPanel)panel.getComponent(0)).add(portSelectorBox);

        textArea = new JTextArea("Please select a port and a client");

        //textArea.setEditable(true);

//        JScrollPane scrollPane = new JScrollPane(textArea);

//        scrollPane.setPreferredSize(new Dimension(400, 200));

        //scrollPane.add(textArea);

        //scrollPane.setVisible(true);

        panel.add(textArea);

        //((JPanel)panel.getComponent(1)).add(scrollPane);

        panel.add(new JPanel());

        ((JPanel)panel.getComponent(2)).setLayout(new FlowLayout());;

        clientSelectorBox = new JComboBox();

        clientSelectorBox.addItem("Select client");

        clientSelectorBox.addItem("Client 1");

        clientSelectorBox.addItem("Client 2");

        ((JPanel)panel.getComponent(2)).add(clientSelectorBox);

        logArea = new JTextArea("Log");

//        JScrollPane scrollPane = new JScrollPane(logArea);

//        scrollPane.setPreferredSize(new Dimension(400, 200));

//        scrollPane.add(logArea);

//        scrollPane.setVisible(true);

        panel.add(logArea);

        panel.add(new JPanel());

        ((JPanel)panel.getComponent(4)).setLayout(new FlowLayout(FlowLayout.LEFT));

        statusField = new JTextField("Status");

        ((JPanel)panel.getComponent(4)).add(statusField);

        sendButton = new JButton("Send");

        sendButton.setEnabled(true);

        sendButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(java.awt.event.ActionEvent e) {

                m.sendMessage(new sendMessage(textArea.getText()));

            }

        });

        ((JPanel)panel.getComponent(2)).add(sendButton);

        startButton = new JButton("Start");

        startButton.setEnabled(true);

        startButton.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(java.awt.event.ActionEvent e) {

               m.startService(new startService((parseInt(portSelectorBox.getSelectedItem().toString())),clientSelectorBox.getSelectedIndex()));

               //sendButton.setEnabled(true);

//                try {

//                    ServerSocket ss=new ServerSocket(parseInt(portSelectorBox.getSelectedItem().toString()));

//                    Socket s=ss.accept();

//

//                } catch (IOException ex) {

//                    ex.printStackTrace();

//                }

            }

        });

        ((JPanel)panel.getComponent(0)).add(startButton);

    }

}