LAB 05 – UDP

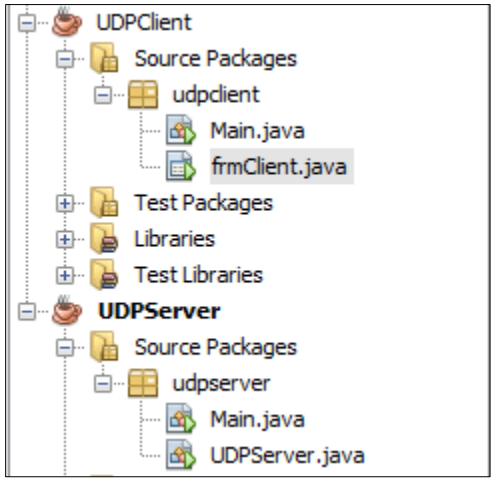
ꕥ

Bài 1: Viết chương trình giao tiếp giữa client và server sử dụng giao thức UDP, thực hiện các chức năng sau:

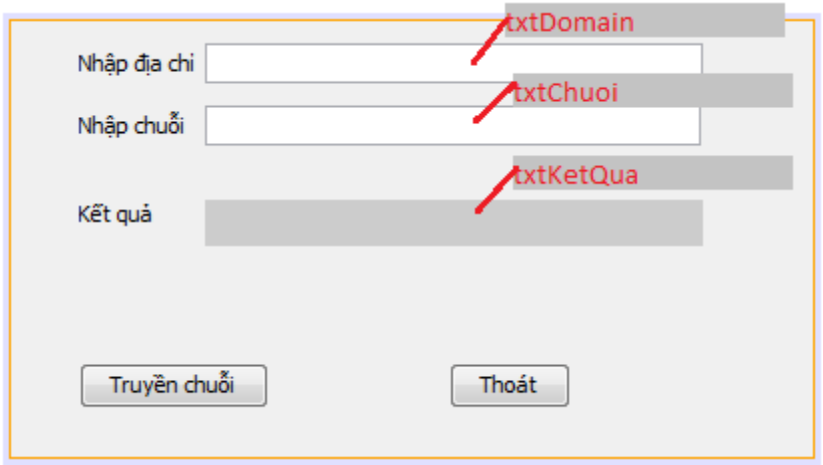
- **Client truyền 1 chuỗi lên server.**
- **Server nhận chuỗi này và chuyển nó thành chữ in hoa sau đó gửi trả kết quả cho client.**
- **Client nhận kết quả rồi sau đó xuất ra màn hình kết quả vừa nhận.**

Yêu cầu: Sử dụng Multithread để server có thể giao tiếp được với nhiều client cùng lúc.

Bước 1: Tạo 2 project mới là UDPServer và UDPClient



Bước 2: Tạo Form frmClient có giao diện như sau:



Xử lý sự kiện cho button Truyền chuỗi.

```
private void btnTruyenChuoiActionPerformed(java.awt.event.ActionEvent evt) {
    byte []sendData;
    DatagramSocket socket;
    try {
        socket = new DatagramSocket();
        String domain=this.txtDomain.getText();
        InetAddress ipServer = InetAddress.getByName(domain);//luu địa chỉ máy server
        int port = 1234;// Sử dụng Port 1234 để giao tiếp với server
        String stSend = this.txtChuoi.getText();//Lấy dữ liệu cần truyền đi
        sendData = stSend.getBytes();//Chuyển dữ liệu thành dạng byte rồi truyền đi
        //DatagramPacket dùng để lưu dữ liệu
        DatagramPacket sendPacket= new DatagramPacket(sendData,sendData.length,ipServer,port);
        socket.send(sendPacket);
        //Nhận chuỗi kết quả từ server
        byte[] buffer=new byte[65507];//độ lớn tối đa của gói tin 65535-(7 byte header của UDP)
        DatagramPacket receivePacket=new DatagramPacket (buffer,buffer.length);
        socket.receive(receivePacket);//Nhận chuỗi kết quả
        txtKetQua.setText(new String(receivePacket.getData()).trim());//Lấy dữ liêuh hiện lên màn hình
        socket.close();//đóng socket
  } catch (Exception ex) {
        JOptionPane.showMessageDialog(this,ex.toString());
    }
}
```

Xử lý sự kiện cho button Thoát

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}
```

Bước 3: Tạo lớp UDPServer như sau:

```
package udpserver;
import java.io.*;
import java.net.*;
public class UDPServer {
	static final int PORT = 1234;//Khai báo Port sử dụng
	private DatagramSocket socket = null;//Khai báo DatagramSocket để lưu kết nối
	public UDPServer(){
		try{
			socket=new DatagramSocket(PORT);
		}catch(Exception e){
			e.printStackTrace();
		}
	}
	public void action(){
		InetAddress host=null;
		int port;
		String chuoi="";//Khai báo biến để lưu chuỗi dữ liệu
		try{
			System.out.println("Server is listening");
			while(true){//vòng lặp chờ
		DatagramPacket packet=receive();//Nhận dữ liệu từ client truyền qua
		host=packet.getAddress();//Lấy thông tin địa chỉ của máy client
		port=packet.getPort();//Lấy thông tin port của máy client
		chuoi=new String(packet.getData()).trim();//Lấy dữ liệu của máy client
				chuoi=chuoi.toUpperCase();//Chuyển thành chữ in hoa
					if(!chuoi.equals(""))
					    send(chuoi,host,port);
			}
		}catch(Exception e){
			e.printStackTrace();
		}finally{
			socket.close();
		}
	}
```

```
	private void send(String chuoi,InetAddress host,int port)throws IOException{
		byte[] buffer=chuoi.getBytes();//chuyển chuỗi truyền thành byte
		   //Sau đó đưa chuỗi truyền vào gói tin gửi đi
		DatagramPacket packet=new DatagramPacket(buffer,buffer.length,host,port);
		socket.send(packet);
	}
	private DatagramPacket receive()throws IOException{
		byte[] buffer=new byte[65507];//Khai báo mảng byte nhận
		DatagramPacket packet=new DatagramPacket (buffer,buffer.length);
		socket.receive(packet);//Nhận dữ liệu
		return packet;
	}
	public static void main(String []args){
		new UDPServer().action();
	}
}
```

Bước 4: Chạy thử ứng dụng:

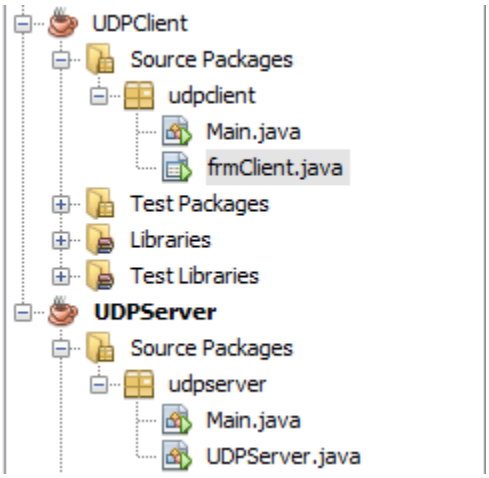
- ✓ Chạy file UDPServer.java
- ✓ Chạy file frmClient.java

Bài 2: Viết chương trình giao tiếp giữa client và server sử dụng giao thức UDP, thực hiện các chức năng sau:

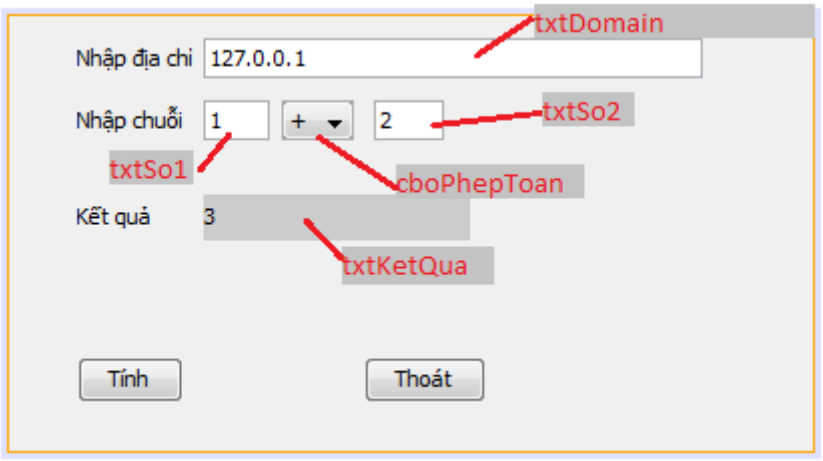
- **Client truyền hai số nguyên và phép toán (cộng, trừ, nhân, chia) lên server.**
- **Server sau khi nhận được thì thực hiện phép toán giữa hai số nguyên và trả kết quả về cho client.**
- **Client nhận lại kết quả và xuất ra màn hình.**

Yêu cầu: Sử dụng Multithread để server có thể giao tiếp được với nhiều client cùng lúc.

Bước 1: Tạo 2 project mới là UDPServer và UDPClient



Bước 2: Tạo Form frmClient có giao diện như sau:



Xử lý sự kiện cho button Tính.

```
private void btnTinhActionPerformed(java.awt.event.ActionEvent evt) {
    byte []sendData;
    DatagramSocket socket;
    int so1= Integer.parseInt(txtSo1.getText());//Lấy số hạng thứ nhất
    int so2=Integer.parseInt(txtSo2.getText());//Lấy số hạng thứ hai
    String pheptoan=cboPhepToan.getSelectedItem().toString();//Lấy phép toán
    try {
        socket = new DatagramSocket();
        String domain=this.txtDomain.getText();
        InetAddress ipServer = InetAddress.getByName(domain);//lưu địa chỉ máy server
        int port = 1234;// Sử dụng Port 1234 để giao tiếp với server
        String stSend = so1+"@"+pheptoan+"@"+so2;//Lấy dữ liệu cần truyền đi
        sendData = stSend.getBytes();//Chuyển dữ liệu thành dạng byte rồi truyền đi
        //DatagramPacket dùng để lưu dữ liệu
        DatagramPacket sendPacket= new DatagramPacket(sendData,sendData.length,ipServer,port);
        socket.send(sendPacket);
        //Nhận chuỗi kết quả từ server
        byte[] buffer=new byte[65507];//độ lớn tối đa của gói tin 65535-(7 byte header của UDP)
        DatagramPacket receivePacket=new DatagramPacket (buffer,buffer.length);
        socket.receive(receivePacket);//Nhận chuỗi kết quả
        txtKetQua.setText(new String(receivePacket.getData()).trim());//Lấy dữ liêuh hiện lên màn hình
        socket.close();//đóng socket
  } catch (Exception ex) {
        JOptionPane.showMessageDialog(this,ex.toString());
    }
}
```

Xử lý sự kiện cho button Thoát

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}
```

Bước 3: Tạo lớp UDPServer như sau:

```
package udpserver;
import java.io.*;
import java.net.*;
import java.util.Scanner;
public class UDPServer {
	static final int PORT = 1234;//Khai báo Port sử dụng
	private DatagramSocket socket = null;//Khai báo DatagramSocket để lưu kết nối
	public UDPServer(){
		try{
			socket=new DatagramSocket(PORT);
		}catch(Exception e){
			e.printStackTrace();
		}
	}
	public void action(){
		InetAddress host=null;
		int port;
		String chuoi="";//Khai báo biến để lưu chuỗi dữ liệu
		try{
			System.out.println("Server is listening");
			while(true){//vòng lặp chờ
		DatagramPacket packet=receive();//Nhận dữ liệu từ client truyền qua
		host=packet.getAddress();//Lấy thông tin địa chỉ của máy client
				port=packet.getPort();//Lấy thông tin port của máy client
		chuoi=new String(packet.getData()).trim();//Lấy dữ liệu của máy client
					if(!chuoi.equals("")){
					    Scanner sc=new Scanner(chuoi);
					    sc.useDelimiter("@");//Cắt chuỗi theo ký tự @
			int so1=sc.nextInt();//Lấy so1 là phần trước chữ @ đầu tiên
			String pheptoan=sc.next();//Phép toán là phần trước chữ @ thứ hai
					int so2=sc.nextInt();//so2 là phần trước chữ @ thứ 3
					if(pheptoan.equals("+"))//Nếu phép toán là phép cộng
						chuoi=(so1+so2)+"";
				else if(pheptoan.equals("-"))//Nếu phép toán là phép trừ
```

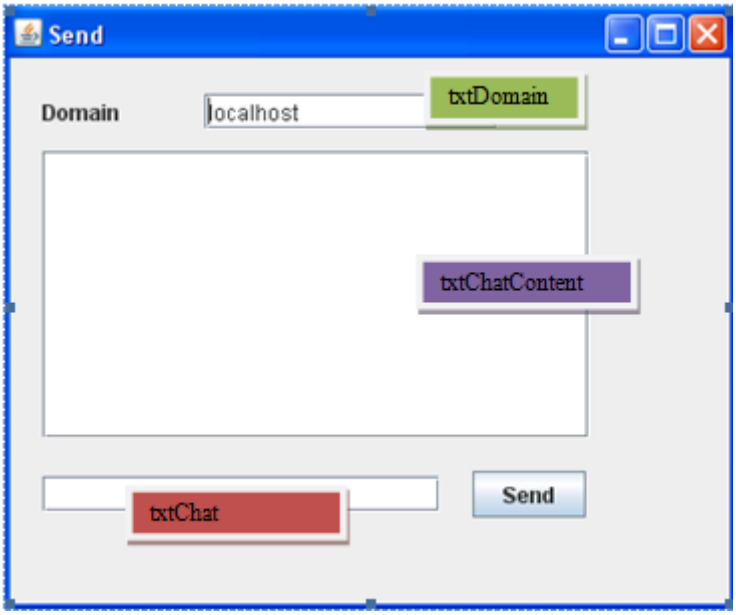
```
                                    chuoi=(so1-so2)+"";
                            else if(pheptoan.equals("*"))//Nếu phép toán là phép nhân
                                    chuoi=(so1*so2)+"";
                            else if(pheptoan.equals("/"))//Nếu phép toán là phép chia
                                    chuoi=((float)so1/so2)+"";
                            send(chuoi,host,port);//Truyền chuỗi trả về cho client
                                }
                            }
                        }catch(Exception e){
                        e.printStackTrace();
                }finally{
                        socket.close();
                }
        }
        private void send(String chuoi,InetAddress host,int port)throws IOException{
                byte[] buffer=chuoi.getBytes();//chuyển chuỗi truyền thành byte
                   //Sau đó đưa chuỗi truyền vào gói tin gửi đi
                DatagramPacket packet=new DatagramPacket(buffer,buffer.length,host,port);
                socket.send(packet);
        }
        private DatagramPacket receive()throws IOException{
                byte[] buffer=new byte[65507];//Khai báo mảng byte nhận
                DatagramPacket packet=new DatagramPacket (buffer,buffer.length);
                socket.receive(packet);//Nhận dữ liệu
                return packet;
        }
        public static void main(String []args){
                new UDPServer().action();
        }
}
```

Bước 4: Chạy thử ứng dụng

- ✓ Chạy file UDPServer.java
- ✓ Chạy file frmClient.java

Bài 3: Viết chương trình cho phép hai máy chat với nhau.

Bước 1: Thiết kế giao diện: lớp Chat.java



Bước 2: Viết code cho button Send của lớp Chat.java

```
byte []sendData;
 boolean ktFinish = false;
DatagramSocket socket;
 String strContent="";
 try {
      socket = new DatagramSocket();
      String domain=this.txtDomain.getText();
      InetAddress ipServer = InetAddress.getByName(domain);
      int port = Chat.PORT;
      String stSend = this.txtChat.getText();
      sendData = stSend.getBytes();
      DatagramPacket sendPacket;
sendPaket= new DatagramPacket(sendData,sendData.length,ipServer,port);
 strContent+="\nGui : " + stSend;
 socket.send(sendPacket); //Begin chat
 socket.close();
} catch (Exception ex) {
      JOptionPane.showMessageDialog(this,ex);
  }
```

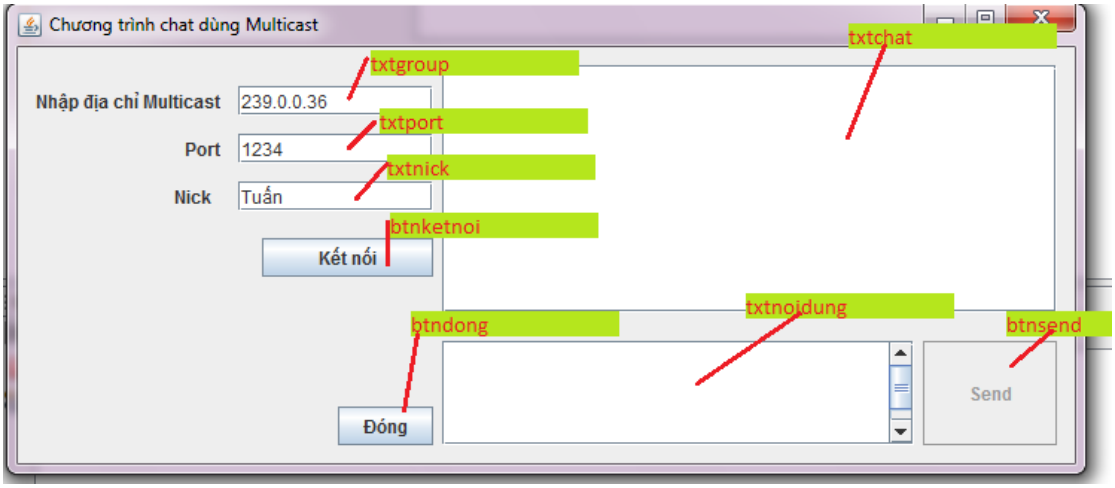
Bước 3: Viết code cho lớp Main.java để tạo ra đối tượng giao diện chat và code xử lý lắng nghe gói dữ liệu chat.

```
        Chat app=new Chat();
        app.setVisible(true);
        //doan nhan du lieu
        DatagramSocket socket;
        String strContent="";
        try {
            byte []buffer = new byte[1024];
            socket = new DatagramSocket(PORT);
            boolean ktFinish=false;
            DatagramPacket receivePacket;
           String stReceive;
            while(ktFinish!=true) {
            receivePacket = new DatagramPacket(buffer,buffer.length);
              socket.receive(receivePacket);
stReceive=new String(receivePacket.getData(),0,receivePacket.getLength());
                strContent=app.getContentChat();
                strContent+="Nhan : " + stReceive;
                app.setContentChat(strContent);
            if (stReceive.equals("end.")||stReceive.equals("end.")) {
             ktFinish = true; }
            }
        }catch(Exception ex){
            JOptionPane.showMessageDialog(null,ex);
        }
```

Bài 4. Viết chương trình cho phép hai máy chat với nhau dùng Multicast.

Bước 1: Tạo JFrame frmChat

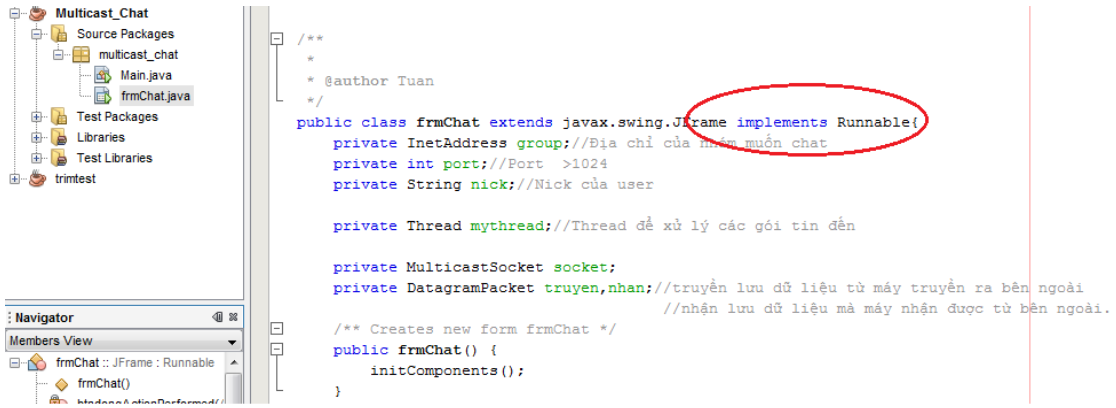
Bước 2: Tạo giao diện như hình sau:



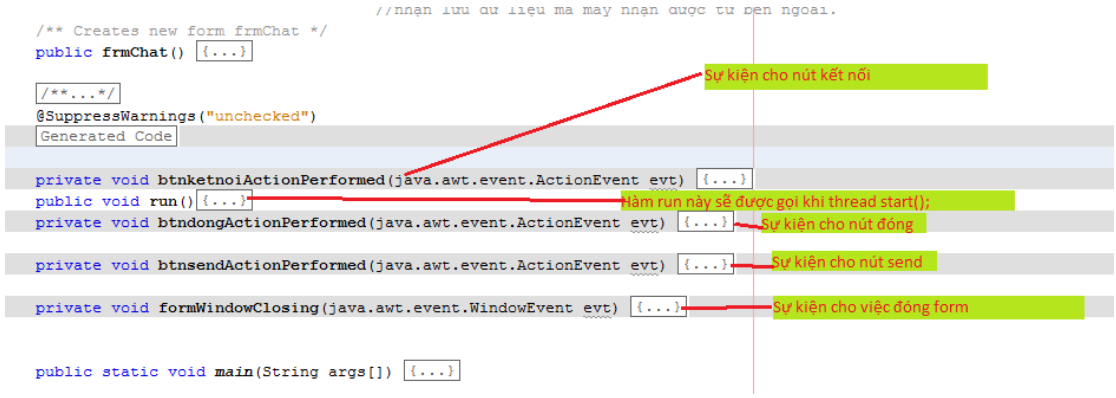
Khai báo các thư viện được sử dụng

```
import java.io.*;
import java.net.*;
import javax.swing.*;
```

Bước 3: Khai báo các thuộc tính của Form



Bước 4: Các hàm trong chương trình



Bước 5: Sự kiện cho nút kết nối

```
private void btnketnoiActionPerformed(java.awt.event.ActionEvent evt) {
      if(btnketnoi.getText().equals("Kết nối")){
  //Khi nhấn nút kết nối
        btnketnoi.setText("Ngắt kết nối");
   //chuyển nút kết nối thành nút ngắt kết nối
        txtchat.setEnabled(true);
        txtnoidung.setEnabled(true);
        btnsend.setEnabled(true);
        txtgroup.setEnabled(false);
        txtport.setEnabled(false);
        txtnick.setEnabled(false);
```

```
        try{
        group= InetAddress.getByName(txtgroup.getText());
        if(group.isMulticastAddress()){
  //Kiểm tra xem địa chỉ nhóm có phải địa chỉ multicast hay không
            nick=txtnick.getText();
            port=Integer.parseInt(txtport.getText());
            if(mythread==null){
  //Tạo ra và thiết lập ban đầu cho các đối tượng mạng
               socket=new MulticastSocket(port);
               socket.setTimeToLive(1);
  //Thiết lập đường đi cho gói tin
               socket.joinGroup(group);
  //Đăng ký với router là chương trình máy mình đăng ký vào nhóm group
               truyen=new DatagramPacket(new byte[1],1,group,port);
               nhan=new DatagramPacket(new byte[65507],65507);
               //Tạo ra thread xử lý dữ liệu truyền đến
               mythread=new Thread(this);
               mythread.start();
  //Bắt đầu nhận dữ liệu - Lúc này hàm run sẽ dược gọi để thực thi
            }
        }else
         JOptionPane.showMessageDialog(null, "Địa chỉ nhập sai rồi!!");
        }catch(Exception e){
                    JOptionPane.showMessageDialog(null,e);
        }
     }else{
              //Sự kiện nhấn nút ngắt kết nối
        txtchat.setEnabled(false);
        txtnoidung.setEnabled(false);
        btnsend.setEnabled(false);
        txtgroup.setEnabled(true);
        txtport.setEnabled(true);
        txtnick.setEnabled(true);
        btnketnoi.setText("Kết nối");
              //chuyển nút ngắt kết nối thành nút kết nối
        if(mythread!=null){
        mythread.interrupt();
                     //dừng việc nhận dữ liệu
        mythread=null;
        try{
           socket.leaveGroup(group);//Ra khỏi group
```

```
        }catch(IOException e){}
        socket.close();
        }
    }
}
```

Bước 5: Hàm run

```
    private void btnketnoiActionPerformed(java.awt.event.ActionEvent evt) {...}
    public void run(){
    try{
        while (!Thread.interrupted()) {//Kiểm tra xem thread có bị ngắt chưa
            nhan.setLength(nhan.getData().length);//thiết lập số byte của buffer
            socket.receive(nhan);//nhận dữ liệu
            String message = new String(nhan.getData(), 0, nhan.getLength(), "UTF8");
            txtchat.append(message + "\n");//hiển thì dữ liệu nhận được lên màn hình
        }
    } catch (IOException e) {//Các thao tác thu dọn bộ nhớ khi có lỗi xảy ra
        if(mythread!=null){
            txtchat.append(e+"\n");
            txtnoidung.setVisible(false);
            this.validate();
            if(mythread!=Thread.currentThread())
                mythread.interrupt();
            mythread=null;
            try{
                socket.leaveGroup(group);
            }catch(IOException ignored){}
            socket.close();
            }
    }

}
```

Bước 6: Sự kiện cho nút đóng

```
    private void btndongActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }
```

Bước 7: Sự kiện cho nút Send

```
private void btnsendActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        byte[] utf=(nick+": "+txtnoidung.getText()).getBytes("UTF8");//Chuyển dữ liệu thành chuỗi byte
        truyen.setData(utf);// gán dữ liệu cho datagrampackage
        truyen.setLength(utf.length);//thiết lập số lượng byte cho buffer
        socket.send(truyen);//bắt đầu truyền dữ liệu đi
        txtnoidung.setText("");//cho nội dung của txtnoidung là rỗng
    }catch(IOException e){//Các xử lý dọn dẹp bộ nhớ khi có lỗi
        if(mythread!=null){
        txtchat.append(e+"\n");
        txtnoidung.setVisible(false);
        this.validate();
        if(mythread!=Thread.currentThread())
            mythread.interrupt();
        mythread=null;
        try{
            socket.leaveGroup(group);
        }catch(IOException ignored){}
        socket.close();
        }
    }
}
```

Bước 8: Sự kiện cho việc đang đóng form

```
private void formWindowClosing(java.awt.event.WindowEvent evt) {
    if(mythread!=null){// các hàm dọn dẹp bộ nhớ
        mythread.interrupt();
        mythread=null;
        try{
            socket.leaveGroup(group);
        }catch(IOException e){}
        socket.close();
        }
}
```

Bước 9: Kiểm tra bằng cách chạy thử chương trình trên nhiều máy mạng LAN chỉ cần thay nick khác nhau cho từng máy. Những máy nào thiết lập cùng group (cùng địa chỉ Multicast) thì có thể chat được với nhau.

Bài 5. Viết chương trình minh hoạt giao thức FTP cho phép hai máy gửi tập tin cho nhau sử dụng UDP Socket.

Bài 6. Viết chương trình cho phép gia nhập vào một địa chỉ multicast. Thực hiện chức năng gởi một thông điệp vào địa chỉ này và nhận các thông điệp được gởi tới địa chỉ này.