**实验四 网络编程基础**

1. **【提交文字和运行界面】**

**实现一个客户端程序和一个服务器程序之间的通信。客户端在控制台输入圆的半径发送给服务器，服务器接受数据并用它来计算面积生成结果返回给客户端。**

半径

客户端

服务器

面积

1. **【提交代码和运行界面】实现多个客户端程序和一个服务器程序之间的通信。客户端在控制台输入圆的半径发送给服务器，服务器对于每个客户端都启动一个线程。该线程接受数据并用它来计算面积生成结果返回给客户端。**
2. **【提交代码和运行界面】实现基于NIO非阻塞Socket通信的客户端程序和服务器程序之间的通信。客户端在控制台输入圆的半径发送给服务器，服务器接受数据并用它来计算面积生成结果返回给客户端。**
3. **【提交填空的代码】编写客户/服务器程序实现一个简单的局域网内的”单词广播系统”。BroadCastWord程序负责将选择的文件在局域网内进行多点广播。加入多点广播地址的各个客户端Receive将收到文件大的内容。知识点包含MulitcastSocket的使用，多点广播的原理，UDP通信的原理和过程。将补充完整的代码并实现功能。**

**import** java.net.\*;

**import**java.awt.\*;

**import**java.awt.event.\*;

**import** java.io.\*;

**import**javax.swing.Timer;

**public class** BroadCastWord **extends** Frame **implements**ActionListener{

**int**port;

InetAddressgroup=**null**;

MulticastSocketsocket=**null**;

Timer time=**null**;

FileDialogopen=**null**;

Button select,开始广播,停止广播;

File file=**null**;

String FileDir=**null**,fileName=**null**;

FileReaderin=**null**;

BufferedReaderbufferIn=**null**;

**int**token=0;

TextArea显示正在播放的内容,显示已经播放的内容;

**public**BroadCastWord(){

**super**("单词广播系统");

select=**new** Button("选择要广播的文件");

开始广播=**new** Button("开始广播");

停止广播=**new** Button("停止广播");

select.addActionListener(**this**);

开始广播.addActionListener(**this**);

停止广播.addActionListener(**this**);

time=**new** Timer(2000,**this**);

open=**new**FileDialog(**this**,"选择要广播的文件",FileDialog.*LOAD*);

显示正在播放的内容=**new**TextArea(10,10);

显示正在播放的内容.setForeground(Color.*blue*);

显示已经播放的内容=**new**TextArea(10,10);

Panel north=**new**Panel();

north.add(select);

north.add(开始广播);

north.add(停止广播);

add(north,BorderLayout.*NORTH*);

Panel center=**new**Panel();

center.setLayout(**new**GridLayout(1,2));

center.add(显示正在播放的内容);

center.add(显示已经播放的内容);

add(center,BorderLayout.*CENTER*);

validate();

**try**{

//指定多点广播目的的端口

port=5000;

/定义本程序的多点广播地址

group=InetAddress.*getByName*("239.255.0.0");

***//创建用于发送和接受数据的MulitcastSocket***

***【代码1】***

socket.setTimeToLive(1);

***//将该socket加入指定的多点广播地址***

***【代码 2】***

}**catch**(Exception e){

System.*out*.println("Error"+e);

}

setBounds(100,50,360,380);

setVisible(**true**);

addWindowListener(**new**WindowAdapter(){

**publicvoid**windowClosing(WindowEvent e)

{

System.*exit*(0) ;

} }

);

}

**publicvoid**actionPerformed(ActionEvent e){

**if**(e.getSource()==select){

显示正在播放的内容.setText(**null**);

open.setVisible(**true**);

fileName=open.getFile();

FileDir=open.getDirectory();

**if**(fileName!=**null**){

time.stop();

file=**new** File(FileDir,fileName);

**try**{

file=**new** File(FileDir,fileName);

in=**new**FileReader(file);

bufferIn=**new**BufferedReader(in);

}**catch**(IOExceptionee){}

}

}

**elseif**(e.getSource()==开始广播){

time.start();

}

**elseif**(e.getSource()==time){

String s=**null**;

**try**{

**if**(token==-1){

file=**new** File(FileDir,fileName);

in=**new**FileReader(file);

bufferIn=**new**BufferedReader(in);

}

s=bufferIn.readLine();

**if**(s!=**null**){

token=0;

显示正在播放的内容.setText("正在广播的内容:\n"+s);

显示已经播放的内容.append(s+"\n");

DatagramPacket packet=**null**;

**byte** data[]=s.getBytes();

***//创建数据报，该数据包的目的地址是多点广播地址，***

***//端口是5000，数据为data中的全部字节***

packet=***【代码3】***

socket.send(packet);

}

**else**{

token=-1;

}

}**catch**(IOExceptionee){}

}

**elseif**(e.getSource()==停止广播){

time.stop();

}

}

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

BroadCastWordbroad=**new**BroadCastWord();

}

}

**import**java.awt.\*;

**import**java.awt.event.\*;

**import** java.net.\*;

**public class** Receive **extends** Frame **implements**Runnable,ActionListener{

**int**port;

InetAddressgroup=**null**;

MulticastSocketsocket=**null**;

Button 开始接收,停止接收;

TextArea显示正在接收的内容,显示已经接收的内容;

Thread thread;

**boolean**停止=**false**;

**public** Receive(){

**super**("定时接收系统");

thread=**new** Thread(**this**);

开始接收=**new** Button("开始接收");

停止接收=**new** Button("停止接收");

开始接收.addActionListener(**this**);

停止接收.addActionListener(**this**);

显示正在接收的内容=**new**TextArea(10,10);

显示正在接收的内容.setForeground(Color.*blue*);

显示已经接收的内容=**new**TextArea(10,10);

Panel north=**new**Panel();

north.add(开始接收);

north.add(停止接收);

add(north,BorderLayout.*NORTH*);

Panel center=**new**Panel();

center.setLayout(**new**GridLayout(1,2));

center.add(显示正在接收的内容);

center.add(显示已经接收的内容);

add(center,BorderLayout.*CENTER*);

validate();

port=5000;

**try**{

group=InetAddress.*getByName*("239.255.0.0");

***//创建用于接受数据的MulitcastSocket***

***【代码4】***

***//将该socket加入指定的多点广播地址***

***【代码5】***

}**catch**(Exception e){

System.*out*.println("Error"+e);

}

setBounds(100,50,360,380);

setVisible(**true**);

addWindowListener(**new**WindowAdapter(){

**publicvoid**windowClosing(WindowEvent e)

{

System.*exit*(0) ;

} }

);

}

**publicvoid**actionPerformed(ActionEvent e){

**if**(e.getSource()==开始接收){

开始接收.setBackground(Color.*blue*);

停止接收.setBackground(Color.*gray*);

***//判断线程是否以及启动，如果线程没有启动或已经结束则重新创建新的线程***

**if**(***【代码6】***){

thread=**new** Thread(**this**);

}

**try**{

System.*out*.println("开始接收");

***//启动线程***

***【代码7】***

停止=**false**;

}**catch**(Exception ee){}

}

**if**(e.getSource()==停止接收){

开始接收.setBackground(Color.*gray*);

停止接收.setBackground(Color.*blue*);

thread.interrupt();

停止=**true**;

}

}

**publicvoid** run(){

**while**(**true**){

**byte** data[]=**newbyte**[8192];

DatagramPacket packet=**null**;

***//创建接收用的DatagramPacket***

***【代码8】***

**try**{

***//socket接收数据***

***【代码9】***

String message=**new**String(packet.getData(),0,packet.getLength());

System.*out*.println(message);

显示正在接收的内容.setText("显示正在接收的内容:\n"+message);

显示已经接收的内容.setText(message+"\n");

}**catch**(Exception e){}

**if**(停止==**true**){

**break**;

}

}

}

**Public static void** main(String[] args) {

**new** Receive();

}

}