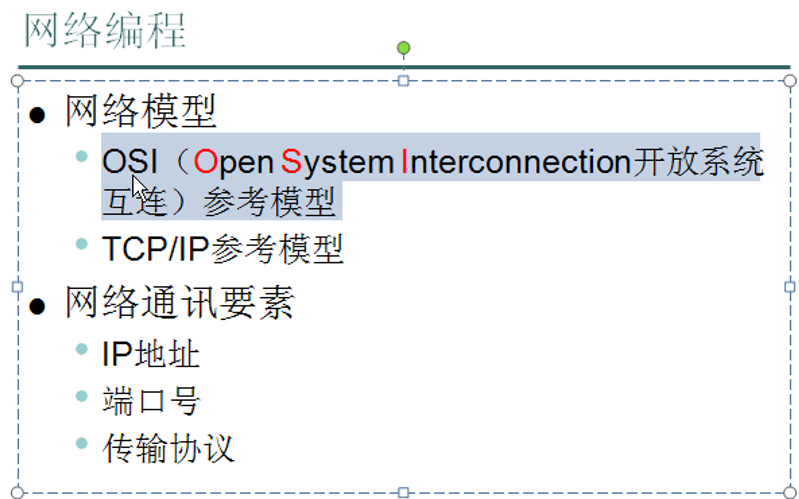
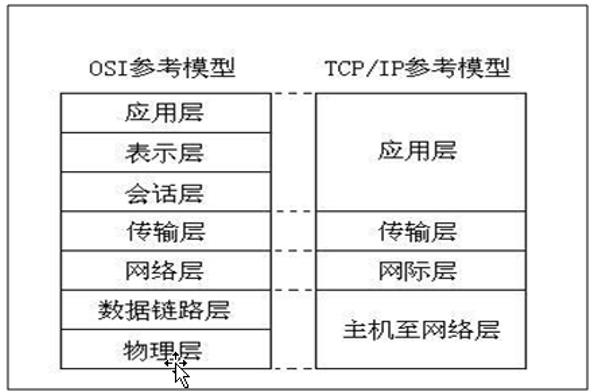
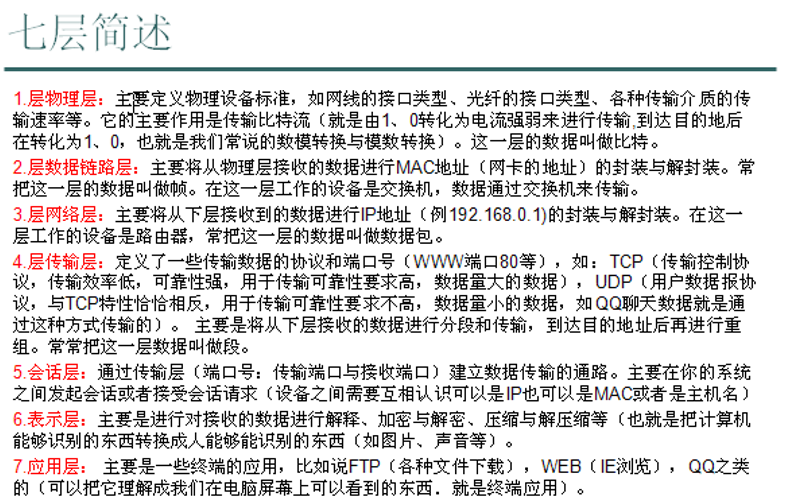
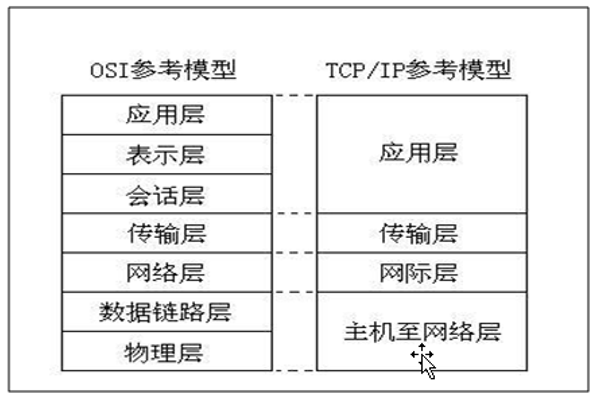
# 网络编程

## 01-网络编程(网络模型概述)



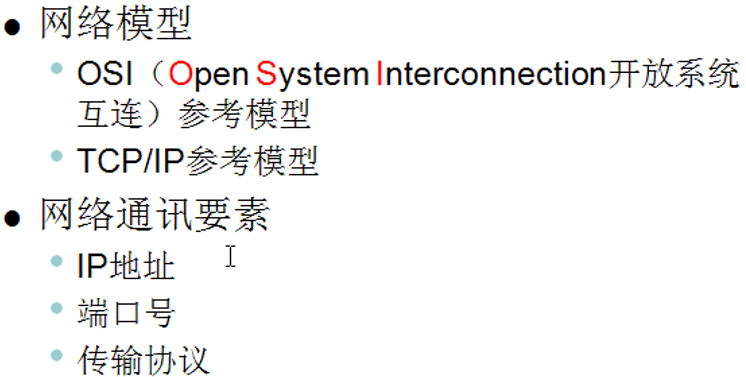


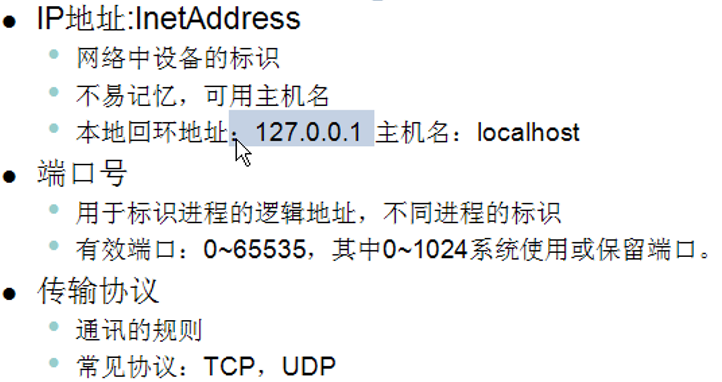




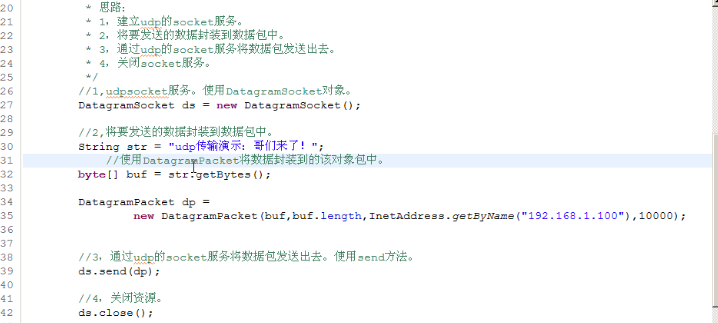
## 02-网络编程(网络要素-IP地址)







## 07-网络编程(UDP协议-发送端)



## 08-网络编程(UDP协议-接收端)

**package** udp;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.net.SocketException;

**public** **class** UDPSendDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

System.*out*.println("send run....");

DatagramSocket ds = **new** DatagramSocket(8888);

String str = "hello";

**byte**[] buf = str.getBytes();

DatagramPacket dp = **new** DatagramPacket(buf, buf.length,InetAddress.*getByName*("127.0.0.1"),10000);

ds.send(dp);

ds.close();

}

}

**package** udp;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**public** **class** UDPReceDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

System.*out*.println("rece run...");

//1、建立UDP接收端的socket服务，因为是要接受数据，所以必须明确一个端口号

DatagramSocket ds = **new** DatagramSocket(10000);

//2、建立数据包

**byte**[] buf =**new** **byte**[1024];

DatagramPacket dp = **new** DatagramPacket(buf, buf.length);

//3、使用接收方法将数据存储到数据包中

ds.receive(dp);//阻塞式的

//4、通过数据对象的方法解析其中的数据，比如：地址、端口、数据内容

String ip = dp.getAddress().getHostAddress();

**int** port = dp.getPort();

String text = **new** String(dp.getData(),0,dp.getLength());

System.*out*.println(ip+":"+port+":"+text);

//5、关闭资源

ds.close();

}

}

## 09-网络编程(UDP协议-聊天程序)

**package** chat;

**import** java.io.IOException;

**import** java.net.DatagramSocket;

**public** **class** ChatDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

DatagramSocket send =**new** DatagramSocket();

DatagramSocket rece =**new** DatagramSocket(100 00);

**new** Thread(**new** Send(send)).start();

**new** Thread(**new** Rece(rece)).start();

}

}

**package** chat;

**import** java.io.BufferedReader;

**import** java.io.InputStreamReader;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**public** **class** Send **implements** Runnable{

**private** DatagramSocket ds;

**public** Send(DatagramSocket ds){

**this**.ds = ds;

}

**public** **void** run(){

**try** {

//读取键盘录入

BufferedReader bufr = **new** BufferedReader(**new** InputStreamReader(System.*in*));

String line = **null**;

**while** ((line=bufr.readLine())!=**null**){

**byte**[] buf = line.getBytes();

DatagramPacket dp = **new** DatagramPacket(buf, buf.length,InetAddress.*getByName*("127.0.0.1"),10000);

ds.send(dp);

**if**("886".equals(line))

**break**;

ds.close();

}

} **catch** (Exception e) {

// **TODO**: handle exception

}

}

}

**package** chat;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**public** **class** Rece **implements** Runnable{

**private** DatagramSocket ds;

**public** Rece(DatagramSocket ds){

**this**.ds = ds;

}

**public** **void** run(){

**try** {

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

**byte**[] buf =**new** **byte**[1024];

DatagramPacket dp = **new** DatagramPacket(buf, buf.length);

ds.receive(dp);

String ip = dp.getAddress().getHostAddress();

**int** port = dp.getPort();

String text = **new** String(dp.getData(),0,dp.getLength());

System.*out*.println(ip+":"+port+":"+text);

**if**(text.equals("886")){

System.*out*.println(ip+"....退出聊天室");

}

}

} **catch** (Exception e) {

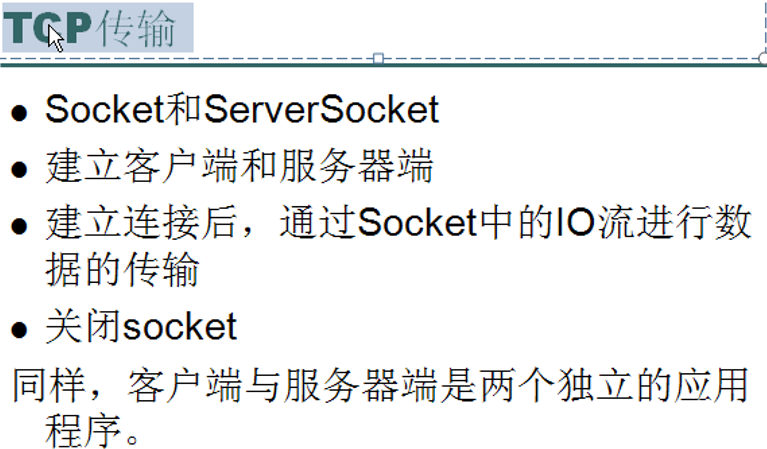
// **TODO**: handle exception

}

}

}

## 10-网络编程(TCP协议-客户端)





**package** p4TCP;

**import** java.io.IOException;

**import** java.io.OutputStream;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** ClientDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\* **@throws** UnknownHostException

\*/

**public** **static** **void** main(String[] args) **throws** UnknownHostException, IOException {

// **TODO** Auto-generated method stub

//客户端发数据到服务端

/\*

\* TCP传输，客户端建立过程

\* 1、创建TCP客户端socket服务。使用Socket对象。

\* 建议该对象以创建就明确目的地，要连接的主机。

\* 2、如果连接建立成功，说明数据传输通道已经建立。

\* 该通道就是socket流，是底层建立好的。既然是流，说明这里既有输入又有输出。

\* 想要输入或者输出流对象，可以找socket来获取。

\* 可以通过getOutputStream（）和getInputStream（）来获取两个字节流。

\* 3、使用输出流，将数据写出。

\* 4、关闭资源。

\*

\*

\*/

//创建客户端socket服务

Socket socket = **new** Socket("192.68.1.1100",10002);

//获取socket流中的输出流

OutputStream out = socket.getOutputStream();

//使用输出流将制定的数据写出去

out.write("tcp演示：哥们儿又回来了！".getBytes());

//关闭资源。

socket.close();

}

}

## 11-网络编程(TCP协议-服务端)

**package** p4TCP;

**import** java.io.IOException;

**import** java.io.OutputStream;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** ClientDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\* **@throws** UnknownHostException

\*/

**public** **static** **void** main(String[] args) **throws** UnknownHostException, IOException {

// **TODO** Auto-generated method stub

//客户端发数据到服务端

/\*

\* TCP传输，客户端建立过程

\* 1、创建TCP客户端socket服务。使用Socket对象。

\* 建议该对象以创建就明确目的地，要连接的主机。

\* 2、如果连接建立成功，说明数据传输通道已经建立。

\* 该通道就是socket流，是底层建立好的。既然是流，说明这里既有输入又有输出。

\* 想要输入或者输出流对象，可以找socket来获取。

\* 可以通过getOutputStream（）和getInputStream（）来获取两个字节流。

\* 3、使用输出流，将数据写出。

\* 4、关闭资源。

\*

\*

\*/

//创建客户端socket服务

Socket socket = **new** Socket("127.0.0.1",10002);

//获取socket流中的输出流

OutputStream out = socket.getOutputStream();

//使用输出流将制定的数据写出去

out.write("tcp演示：哥们儿又回来了！".getBytes());

//关闭资源。

socket.close();

}

}

**package** p4TCP;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** ServerDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

//服务器接受客户端发送过来的数据，并打印在控制台上

/\*

\* 建立TCP服务端的思路：

\* 1、创建服务端socket服务，通过ServerSocket对象

\* 2、服务端必须对外提供一个端口，柔则客户端无法连接。

\* 3、获取链接过来的客户端对象

\* 4、通过客户端对象获取Socket流读取客户端发来的数据并打印在控制台上

\* 5、关闭资源。关客户端、关服务端。

\*

\*/

//创建服务器对象

ServerSocket ss = **new** ServerSocket(10002);

//获取连接过来的客户端对象

Socket s = ss.accept();

String ip = s.getInetAddress().getHostAddress();

//通过Socket对象获取输入流，尧都区客户端发来的数据

InputStream in = s.getInputStream();

**byte**[] buf = **new** **byte**[1024];

**int** len = in.read(buf);

String text = **new** String (buf,0,len);

System.*out*.println(ip+":"+text);

s.close();

ss.close();

}

}

## 12-网络编程(TCP协议-服务端和客户端交互)

**package** p4TCP;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.OutputStream;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** ClientDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\* **@throws** UnknownHostException

\*/

**public** **static** **void** main(String[] args) **throws** UnknownHostException, IOException {

// **TODO** Auto-generated method stub

//客户端发数据到服务端

/\*

\* TCP传输，客户端建立过程

\* 1、创建TCP客户端socket服务。使用Socket对象。

\* 建议该对象以创建就明确目的地，要连接的主机。

\* 2、如果连接建立成功，说明数据传输通道已经建立。

\* 该通道就是socket流，是底层建立好的。既然是流，说明这里既有输入又有输出。

\* 想要输入或者输出流对象，可以找socket来获取。

\* 可以通过getOutputStream（）和getInputStream（）来获取两个字节流。

\* 3、使用输出流，将数据写出。

\* 4、关闭资源。

\*

\*

\*/

//创建客户端socket服务

Socket socket = **new** Socket("127.0.0.1",10002);

//获取socket流中的输出流

OutputStream out = socket.getOutputStream();

//使用输出流将制定的数据写出去

out.write("tcp演示：哥们儿又回来了！".getBytes());

//读取服务端返回的数据，使用Socket读取流。

InputStream in = socket.getInputStream();

**byte**[] buf =**new** **byte**[1024];

**int** len =in.read(buf);

String text = **new** String(buf,0,len);

System.*out*.println(text);

//关闭资源。

socket.close();

}

}

**package** p4TCP;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.OutputStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** ServerDemo {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

//服务器接受客户端发送过来的数据，并打印在控制台上

/\*

\* 建立TCP服务端的思路：

\* 1、创建服务端socket服务，通过ServerSocket对象

\* 2、服务端必须对外提供一个端口，柔则客户端无法连接。

\* 3、获取链接过来的客户端对象

\* 4、通过客户端对象获取Socket流读取客户端发来的数据并打印在控制台上

\* 5、关闭资源。关客户端、关服务端。

\*

\*/

//创建服务器对象

ServerSocket ss = **new** ServerSocket(10002);

//获取连接过来的客户端对象

Socket s = ss.accept();//阻塞式

String ip = s.getInetAddress().getHostAddress();

//通过Socket对象获取输入流，尧都区客户端发来的数据

InputStream in = s.getInputStream();

**byte**[] buf = **new** **byte**[1024];

**int** len = in.read(buf);

String text = **new** String (buf,0,len);

System.*out*.println(ip+":"+text);

//使用客户端Socket对象的输出流留给客户端返回数据

OutputStream out = s.getOutputStream();

out.write("收到".getBytes());

s.close();

ss.close();

}

}

## 13-网络编程(TCP协议-练习-文本转换客户端)

**package** P5TCPtest;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintWriter;

**import** java.net.Socket;

**public** **class** TransClient {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

/\*

\* 思路：

\* 客户端：

\* 1、需要先有socket端点

\* 2、客户端的数据源：键盘

\* 3、客户端的目的：socket。

\* 4、接收服务端的数据，源：socket

\* 5、将数据打印出来：目的：控制台

\* 6、在这些流中操作的数据，都是文本数据。

\*

\*

\* 转换客户端：

\* 1、创建socket客户端对象

\* 2、获取键盘录入

\* 3、将录入的信息发送给socket输出流

\*/

//1、创建socket客户端对象

Socket s = **new** Socket("127.0.0.1",10004);

//2、获取键盘录入

BufferedReader bufr = **new** BufferedReader(**new** InputStreamReader(System.*in*));

//3、socket输出流

// new BufferedWriter(new OutputStreamWriter(s.getOutputStream()));

PrintWriter out = **new** PrintWriter(s.getOutputStream(),**true**);

//4、socket输入流，读取服务器端返回的大写数据

BufferedReader bufIn = **new** BufferedReader(**new** InputStreamReader(s.getInputStream()));

String line = **null**;

**while**((line=bufr.readLine())!=**null**){

**if**("over".equals(line))

**break**;

out.println(line);

//读取服务端发回的一行大写数据

String upperStr = bufIn.readLine();

System.*out*.println(upperStr);

}

s.close();

}

}

## 14-网络编程(TCP协议-练习-文本转换服务端)

**package** P5TCPtest;

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintWriter;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** TransServer {

/\*\*

\* **@param** args

\* **@throws** IOException

\*/

**public** **static** **void** main(String[] args) **throws** IOException {

/\*

\* 转换服务端。

\* 分析：

\* 1、serversocket服务

\* 2、获取socket对象

\* 3、源：socket，读取客户端发过来需要转换的数据。

\* 4、目的：显示在控制台上。

\* 5、将数据转成大写发给客户端。

\*/

//1、

ServerSocket ss =**new** ServerSocket(10004);

//2、获取socket对象。

Socket s = ss.accept();

//获取ip

String ip = s.getInetAddress().getHostAddress();

System.*out*.println(ip+".....connected");

//3、获取socket读取流，并装饰

BufferedReader bufIn = **new** BufferedReader(**new** InputStreamReader(s.getInputStream()));

//4、获取socket的输出流，并装饰。

PrintWriter out = **new** PrintWriter(s.getOutputStream(),**true**);

String line = **null**;

**while**((line=bufIn.readLine())!=**null**){

System.*out*.println(line);

out.print(line.toUpperCase());

}

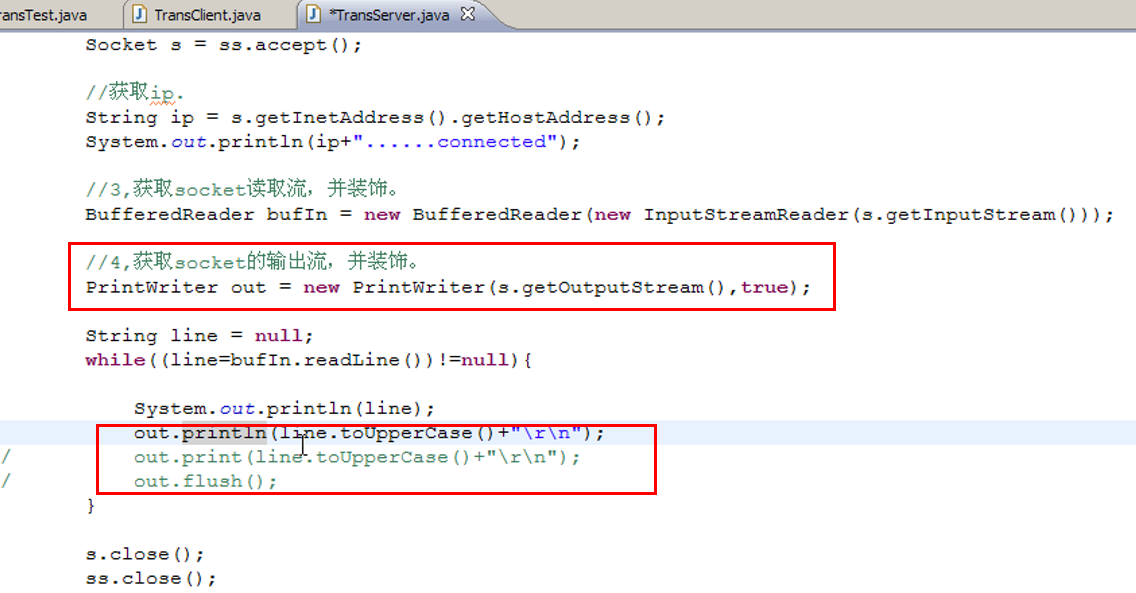
s.close();

ss.close();

}

}

## 15-网络编程(TCP协议-练习-常见问题)



## 16-网络编程(TCP协议-练习-上传文本文件)

## 20-网络编程(常见客户端和服务端)

服务器：处理请求，给予应答