**《面向对象程序设计(Java)》**

**实验报告**

姓 名 石家晖

学 号 2019081238

学 院 软件工程学院

专 业 软件工程

年 级 2019级

指导教师 文立玉

**成都信息工程大学软件工程学院**

**2020年9月**

# 实验9：

1. **实验目的**
   1. 理解Java网络模型。
   2. 掌握InetAddress类的使用。
   3. 理解Socket的相关概念、原理。
   4. 掌握Java Socket编程，学会创建Java网络服务器和网络客户机。
   5. 理解基于UDP协议的Socket的相关概念、原理。
   6. 掌握基于UDP协议的Socket编程的步骤及方法。
   7. 巩固Java GUI 编程和数据流的使用。
2. **实验要求**
3. 实验在2学时内完成。
4. 1人1组独立完成。
5. **实验设备与环境**

JDK13, NEOVIM，Linux

1. **设计思路与具体实现**

1.

(1)设计思路：

基于 TCP 的 Socket 编程。使用套接字编写客户—服务器程序，实现客户 —服务器交互计算。客户将三角形 3 个边的长度发给服务器，服务器把计算 出的三角形的面积返回给客户。

(2)实验步骤

服务器端：Server.java

(a). 调用ServerSocket(int port)创建一个服务器端套接字，并绑定到指定端口上；

(b). 调用accept()，监听连接请求，如果客户端请求连接，则接受连接，返回通信套接字；

(c). 调用Socket类的getOutputStream()和getInputStream获取输出流和输入流，使用输入流获取客户端传来的3条边的长度；

(d). 计算面积；

(e). 将面积通过输出流传递到客户端；

(f). 最后关闭通信套接字。

客户端Client.java

(a). 创建界面，在类的构造器中调用Socket()创建一个流套接字；

(b). 进行事件处理：按钮对象调用addActionListener注册监听，重写actionPerformed方法进行事件处理的具体操作

(c). 在事件处理过程中，客户端Socket连接到服务器端，调用Socket类的getOutputStream()和getInputStream获取输出流和输入流，开始网络数据的发送和接收。将三边传到服务器端，从服务器端获取计算出来的面积。最后关闭通信套接字。

(3) 关键代码

try {

ServerSocket server = null;

try {

server = new ServerSocket(4700);

} catch (Exception e) {

e.printStackTrace();

}

Socket socket = null;

try {

socket = server.accept();

} catch (Exception e) {

e.printStackTrace();

}

double d = 0;

double a[] = new double[NUM];

DataInputStream dataInputStream = new DataInputStream(socket.getInputStream());

DataOutputStream dataOutputStream = new DataOutputStream(socket.getOutputStream());

for (int i = 0; i < a.length; i++) {

d = dataInputStream.readDouble();

a[i] = d;

System.out.println(a[i]);

}

r = (a[0]+a[1]+a[2])/2;

area = Math.sqrt(r\*(r-a[0])\*(r-a[1])\*(r-a[2]));

System.out.println("面积："+area);

dataOutputStream.writeDouble(area);

dataInputStream.close();

dataOutputStream.close();

socket.close();//关闭Socket server.close();//关闭ServerSocket

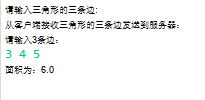
(4) 程序测试过程

①分别编译运行客户端和服务端

②在客户端输入测试样例

③检查服务端是否返回客户端正确结果

(5) 运行结果（运行效果）

2.

(1)设计思路：

基于UDP的Socket编程。编写一个基于UDP协议的Socket网络通信应用程序，实现如下功能：

①客户端能够发送任意的信息给服务器端。

②服务器端将收到的字符串变成大写后返还给客户端。

(2)实验步骤

接收端

(a). 调用DatagramSocket(int port)创建一个数据报套接字，并绑定到指定端口上；

(b). 调用DatagramPacket(byte[] buf, int length)，建立一个字节数组以接收UDP包；

(c). 调用DatagramSocket类的receive()，接收UDP包；

(d). 最后关闭数据报套接字。

发送端

(a). 调用DatagramSocket()创建一个数据报套接字；

(b). 调用DatagramPacket(byte[] buf, int offset, int length, InetAddress address, int port)，建立要发送的UDP包；

(c). 调用DatagramSocket类的send()，发送UDP包；

(d). 最后关闭数据报套接字

(3) 关键代码

try {

System.out.println("服务端已经成功运行，正在等待接受客户端所传输的信息：");

DatagramSocket a = new DatagramSocket(4800);

DatagramPacket receivePacket = new DatagramPacket(receive,receive.length);

a.receive(receivePacket);

String s=new String(receivePacket.getData(),0,receivePacket.getLength());

System.out.println("客户端发送的字符串为："+s);

InetAddress IPAddress = receivePacket.getAddress();

int port = receivePacket.getPort();

String str = s.toUpperCase();

sd=str.getBytes();

DatagramPacket sendPacket=new DatagramPacket(sd,sd.length,IPAddress,port);

a.send(sendPacket);

}catch(Exception e) {

e.printStackTrace();

}

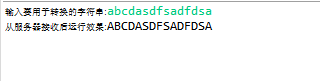
(4) 程序测试过程

①分别编译运行客户端和服务端

②在客户端输入测试样例

③检查服务端是否返回客户端正确结果

(5) 运行结果（运行效果）



1. **实验总结**

通过本次实验我理解了JAVA计算机网络编程的概念、TCP/IP协议规范，理解了理解域名与DNS的概念。在关于服务端与客服端之间的联系，客服端的输入，就对应于服务端的输出，反之类似，了解其大致原理之后，运用了BufferedReader和PrintWr来对其输入输出进行处理，方便了字符的读与写。DatagramSocket以及DatagramPacket的联合用法也可以让socket进行服务端与客服端的联系，当然用完方法之后，一定要记得close()。

1. **附录（程序代码）**

1.

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.net.Socket;

import java.util.Scanner;

public class Client {

public static void main (String args[]) {

int[] Triangle = new int[3];

Scanner scan = new Scanner (System.in);

Socket socket = null;

DataInputStream input = null;

DataOutputStream output = null;

System.out.println ("输入边A：");

Triangle[0] = scan.nextInt();

System.out.println ("输入边B：");

Triangle[1] = scan.nextInt();

System.out.println ("输入边C：");

Triangle[2] = scan.nextInt();

try {

socket = new Socket ("127.0.0.1", 8080); //接入端口

input = new DataInputStream (socket.getInputStream() ); //接收数据

output = new DataOutputStream (socket.getOutputStream() ); //发送数据

for (int i = 0; i < 3; i++) {

output.writeInt (Triangle[i]);

Thread.sleep (1000);

}

double result = input.readDouble(); //读取信息

System.out.println ("三角形面积：" + result);

} catch (Exception e) {

System.out.println ("连接失败！");

}

}

}

import java.io.\*;

import java.net.\*;

public class Server {

public static void main (String[] args) {

int[] Triangle = new int[3];

ServerSocket serversocket = null;

try {

serversocket = new ServerSocket (8080); //端口接入

} catch (IOException e) {

System.out.println (e);

}

try {

System.out.println ("等待客户端连接");

Socket socket = serversocket.accept(); //等待客户端请求连接

DataInputStream input = new DataInputStream (socket.getInputStream() ); //接收数据

DataOutputStream output = new DataOutputStream (socket.getOutputStream() ); //发送数据

for (int i = 0; i < 3; i++) {

Triangle[i] = input.readInt(); //传入接收的数据

Thread.sleep (1000);

}

output.writeDouble (Result (Triangle) );

output.flush();

socket.close();

} catch (Exception e) {

System.out.println (e);

}

}

public static double Result (int []A) {

double ans = 0;

double p = (A[0] + A[1] + A[2]) / 2;

try {

ans = Math.sqrt (p \* (p - A[0]) \* (p - A[1]) \* (p - A[2]) );

} catch (Exception e) {

System.out.println ("输入数据有误！");

}

finally {

return ans;

}

}

}

2.

import java.io.\*;

import java.net.\*;

import java.util.Scanner;

public class UPDClient {

public static void main (String args[]) {

Scanner scan = new Scanner (System.in);

try {

BufferedReader inFromUser = new BufferedReader (new InputStreamReader (System.in) );

DatagramSocket socket = new DatagramSocket();

InetAddress IPAddress = InetAddress.getByName ("127.0.0.1"); //ip地址

byte[] sendData;

byte[] receiveData = new byte[1024];

System.out.print ("输入要发送的字符串：");

String sentence = inFromUser.readLine();//存放输入的句子

sendData = sentence.getBytes(); //转换为byte类型

DatagramPacket sendPacket = new DatagramPacket (sendData, sendData.length, IPAddress, 9876); //发送包

socket.send (sendPacket);

DatagramPacket receivePacket = new DatagramPacket (receiveData, receiveData.length); //接收包

socket.receive (receivePacket);

String modifiedSentence = new String (receivePacket.getData(), 0, receivePacket.getLength() ); //把接收到的数据转换为字符串

System.out.println ("服务器端返回的信息：" + modifiedSentence);

} catch (Exception e) {

System.out.println (e);

}

}

}

import java.net.\*;

public class UPDServer {

public static void main (String args[]) {

byte[] receiveData = new byte[1024];

byte[] sendData;

try {

System.out.println ("服务器在等待接收数据包...");

DatagramSocket socket = new DatagramSocket (9876);

DatagramPacket receivePacket = new DatagramPacket (receiveData, receiveData.length);

socket.receive (receivePacket); //接收客户端传入的包

String sentence = new String (receivePacket.getData(), 0, receivePacket.getLength() ); //转换为字符串

System.out.println ("客户端说：" + sentence);

InetAddress IPAddress = receivePacket.getAddress();

int port = receivePacket.getPort();//获取客户端的端口

String capitalizedSentence = sentence.toUpperCase();//将接收到的字符串转换为大写

sendData = capitalizedSentence.getBytes(); //转换为byte型

DatagramPacket sendPacket = new DatagramPacket (sendData, sendData.length, IPAddress, port);

socket.send (sendPacket);

} catch (Exception e) {

System.out.println (e);

}

}

}