第四部分
综合应用实验

22 实验 17：网络应用

22.1 实验目的

通过实践体会网络套接字是基于 TCP 协议的有连接通信，套接字连接就是客户端的套接字对象和服务器端的套接字对象通过输人输出流连接在一起。熟悉怎样在服务器上建立 ServerSocket 对象，并让 serversocket 对象负责等待客户端请求建立套接字连接。熟悉客户端怎样建立 Socket 对象、向服务器发出套接字连接请求。

22.2 实验要求

客户端和服务器建立套接字连接后，客户将如下格式的账单发送给服务器：

"房租:2189元　　水费:112.9元　　电费:569元　　物业费:832元"

服务器返回给客户的信息是：

您的账单："房租:2189元　　水费:112.9元　　电费:569元　　物业费:832元　　总额:3702.9元"

```
MBP-One:exp yuw$ java ServerItem
正在等待客户
客户的地址:/127.0.0.1
正在监听
正在等待客户
2189.0
112.9
569.0
832.0
```

```
MBP-One:exp yuw$ java ClientItem
输入服务器的IP:127.0.0.1
输入端口号:4331
输入账单:
房租:2189元　　水费:112.9元　　电费:569元　　物业费:832元
您的账单:
账单房租:2189元　　水费:112.9元　　电费:569元　　物业费:832元
总额:3702.9元
```

22.3 程序模板

请按模板要求，将【代码】替换为 Java 程序。

ClientItem.java

```
import java.io.*;
import java.net.*;
import java.util.*;
public class ClientItem {
  public static void main(String args[]) {
    Scanner scanner = new Scanner(System.in);
    Socket clientSocket = null;
    DataInputStream inData = null;
    DataOutputStream outData = null;
    Thread thread;
    Read read = null;
    try{
      clientSocket = new Socket();
      read = new Read();
      thread = new Thread(read);    //负责读取信息的线程
      System.out.print("输入服务器的 IP:");
      String IP = scanner.nextLine();
      System.out.print("输入端口号:");
      int port = scanner.nextInt();
      String enter = scanner.nextLine(); //消耗回车
      if(clientSocket.isConnected()){}
      else{
        InetAddress address = InetAddress.getByName(IP);
        InetSocketAddress socketAddress=new InetSocketAddress(address,port);
        clientSocket.connect(socketAddress);
        InputStream in =【代码 1】//clientSocket调用 getInputStream()返回输入流
        OutputStream out =【代码 2】//clientSocket调用 getOutputStream()返回输出流
        inData = new DataInputStream(in);
        outData = new DataOutputStream(out);
        read.setDataInputStream(inData);
        read.setDataOutputStream(outData);
        thread.start();    //启动负责读信息的线程
      }
    }
    catch(Exception e) {
      System.out.println("服务器已断开"+e);
    }
  }
}
class Read implements Runnable {
  Scanner scanner = new Scanner(System.in);
  DataInputStream in;
  DataOutputStream out;
  public void setDataInputStream(DataInputStream in) {
    this.in = in;
  }
  public void setDataOutputStream(DataOutputStream out) {
    this.out = out;
  }
  public void run() {
    System.out.println("输入账单:");
    String content = scanner.nextLine();
    try{
      out.writeUTF("账单"+content);
      String str = in.readUTF();
      System.out.println(str);
      str = in.readUTF();
      System.out.println(str);
      str = in.readUTF();
      System.out.println(str);
    }
    catch(Exception e) {}
  }
}
```

ServerItem.java

```
import java.io.*;
import java.net.*;
import java.util.*;
public class ServerItem {
  public static void main(String args[]) {
    ServerSocket server = null;
    ServerThread thread;
    Socket you = null;
    while(true) {
      try{
        server = 【代码 3】//创建在端口 4331上负责监听的 ServerSocket对象
      }
      catch(IOException e1) {
        System.out.println("正在监听");
      }
      try{
        System.out.println("正在等待客户");
        you = 【代码 4】 // server调用 accept()返回和客户端相连接的 Socket对象
        System.out.println("客户的地址:"+you.getInetAddress());
      }
      catch (IOException e) {
        System.out.println(""+e);
      }
      if(you!=null) {
        new ServerThread(you).start();
      }
    }
  }
}
class ServerThread extends Thread {
  Socket socket;
  DataInputStream in = null;
  DataOutputStream out = null;
  ServerThread(Socket t) {
    socket = t;
    try {
      out = new DataOutputStream(socket.getOutputStream());
      in = new DataInputStream(socket.getInputStream());
    }
    catch(IOException e) {}
  }
  public void run() {
    try{
      String item = in.readUTF();
      Scanner scanner = new Scanner(item);
      scanner.useDelimiter("[^0123456789.]+");
      if(item.startsWith("账单")) {
        double sum = 0;
        while(scanner.hasNext()){
          try{
            double price = scanner.nextDouble();
            sum = sum + price;
            System.out.println(price);
          }
          catch(InputMismatchException exp){
            String t = scanner.next();
          }
        }
        out.writeUTF("您的账单:");
        out.writeUTF(item);
        out.writeUTF("总额:"+sum+"元");
      }
    }
    catch(Exception exp){}
  }
```

```
}
```

22.4 实验指导

可以使用 Socket 类不带参数的构造方法 **public socket()**创建一个套接字对象，该对象不请求任何连接。该对象再调用 **public void connect(SocketAddress endpoint) throws IOException** 请求和参数 SocketAddress 指定地址的套接字建立连接。为了使用 connect 方法，可以使用 SocketAddress 的子类 InetSocketAddress 创建一个对象，InetSocketAddress 的构造方法是：**publicInetSocketAddress(InetAddress addr, int port)。**

22.5 扩展练习

22.5.1 改进服务器程序计算体积

改进服务器端程序，使得用户还可以发送如下格式的货品明细给服务器：

货品　　宽 90厘米　　高 69厘米　　长 156厘米

服务器返回给客户的信息是：

货品　　宽 90厘米　　高 69厘米　　长 156厘米

体积：968760立方厘米

22.5.2 解析 URL

输入下面的 Java 应用程序，运行程序，分析运行结果。

ParseURL.java

```
import java.net.*;
import java.io.*;
public class ParseURL {
  public static void main(String[] args) throws Exception {
    URL aURL = new URL("http://localhost:200/text.txt#BOTTOM");
    System.out.println(aURL);
    System.out.println("protocol = " + aURL.getProtocol());
    System.out.println("host = " + aURL.getHost());
    System.out.println("filename = " + aURL.getFile());
    System.out.println("port = " + aURL.getPort());
    System.out.println("ref = " + aURL.getRef());
  }
}
```

23 独立实验任务 4（需要提交实验报告）

23.1 实验目的

熟悉 Java 综合应用程序的开发。

23.2 实验任务

编写一个 Java 应用程序，实现图形界面多人聊天室（多线程实现），要求聊天室窗口标题是“欢迎使用 XXX 聊天室应用”，其中 XXX 是自己的班级姓名学号，如“软件 171 张三 1234”。