# 1．TCP编程

## 1.1使用socket和ServerSocket实现udp网络编程

### 1.1.1客户端

|  |
| --- |
| import java.io.IOException;  import java.io.OutputStream;  import java.io.PrintWriter;  import java.net.Socket;  import java.net.UnknownHostException;  /\*  \* 客户端  \*/  public class Client {  /\*\*  \* @param args  \*/  public static void main(String[] args) {  Socket s = null;  try {  // 向服务器发起连接  // 192.168.13.221服务器的ip 9000  // 服务器的端口  s = new Socket("192.168.13.221", 9000);  OutputStream out = s.getOutputStream();  PrintWriter writer = new PrintWriter(out);  writer.write("hello server");  writer.flush();  } catch (UnknownHostException e) {  e.printStackTrace();  } catch (IOException e) {  e.printStackTrace();  } finally {  if (s != null) {  try {  s.close();  } catch (IOException e) {  e.printStackTrace();  }  }  }  }  } |

### 1.1.2服务端

|  |
| --- |
| import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStream;  import java.io.InputStreamReader;  import java.net.ServerSocket;  import java.net.Socket;  /\*  \* 先启动服务器 （ip 固定，不能经常变动:port固定，不能经常变动），启动客户端 （客户端的ip随机，端口也随机）  \* 服务器作为接收消息端  \*/  public class Server {  public static void main(String[] args) {  ServerSocket ss = null;  Socket s = null;  try {  ss = new ServerSocket(9000);// 服务器开放的端口是9000，ip :  // 192.168.13.221;  while (true) {  s = ss.accept();// 开始监听客户端的连接,阻塞状态  InputStream in = s.getInputStream();  InputStreamReader reader = new InputStreamReader(in);  BufferedReader br = new BufferedReader(reader);  String str = br.readLine();  System.out.println(str);  }  } catch (IOException e) {  e.printStackTrace();  } finally {  try {  if (s != null) {  s.close();  }  } catch (IOException e) {  e.printStackTrace();  }  }  }  } |

注意在客户端与服务端交互的过程中不能关闭流。

# 2．UDP编程

## 2.1使用DatagramSocket

### 2.1.1 客户端

|  |
| --- |
| package com.net.object.copy.copy;  import java.io.IOException;  import java.net.DatagramPacket;  import java.net.DatagramSocket;  import java.net.InetSocketAddress;  import java.net.SocketException;  public class UDPClient {  /\*\*  \* @param args  \*/  public static void main(String[] args) {  try {  // 用于连接  DatagramSocket s = new DatagramSocket();  String str = "hello server";  byte[] bs = str.getBytes();  // 数据包 用于传输数据  DatagramPacket dp = new DatagramPacket(bs, 0, bs.length,  new InetSocketAddress("192.168.13.221", 6800));  // 发送给服务器  s.send(dp);  s.close();  } catch (SocketException e) {  e.printStackTrace();  } catch (IOException e) {  e.printStackTrace();  }  }  } |

### 2.1.2服务端

|  |
| --- |
| package com.net.object.copy.copy;  import java.io.IOException;  import java.net.DatagramPacket;  import java.net.DatagramSocket;  import java.net.SocketException;  public class UDPServer {  /\*\*  \* @param args  \*/  public static void main(String[] args) {  try {  DatagramSocket s = new DatagramSocket(6800);  byte[] bs = new byte[20];  DatagramPacket dp = new DatagramPacket(bs, 0, bs.length);  s.receive(dp);  String str = new String(bs, 0, dp.getLength());  System.out.println(str);  s.close();  } catch (SocketException e) {  e.printStackTrace();  } catch (IOException e) {  e.printStackTrace();  }  }  } |

# 3. URL编程

|  |
| --- |
| package com.net.object.copy.copy;  import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStream;  import java.io.InputStreamReader;  import java.net.MalformedURLException;  import java.net.URL;  import java.net.URLConnection;  public class URLTest {  /\*\*  \* @param args  \*/  public static void main(String[] args) {  try {  URL url = new URL("http://www.baidu.com:80/index.html");  URLConnection conn = url.openConnection();  InputStream in = conn.getInputStream();  BufferedReader reader = new BufferedReader(  new InputStreamReader(in));  String s = null;  while ((s = reader.readLine()) != null) {  System.out.println(s);  }  in.close();  } catch (MalformedURLException e) {  e.printStackTrace();  } catch (IOException e) {  e.printStackTrace();  }  }  } |