ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

WLAN Chat Software Based on Java Socket

Dongyang Zhang[1, a], Tiemin Jiang[1, b] and Huawei Mei[1, c]

[1]School of Control and Computer Engineering, North China Electric Power University, Hebei 071003, China.

[a]backyla@qq.com, [b]2182221003@ncepu.edu.cn, [c]593401232@qq.com

Abstract

With the development of computer technology and the continuous popularization of the network, the development of network applications is becoming more and more common, so the development of network applications is particularly important. In this article, the knowledge of TCP/IP programming interface Java Socket under Android is introduced. The classification of sockets and the connection-oriented socket communication mechanism are introduced. The network WIFI is realized by Socket in Android environment. The basics of communication and how to create connection-oriented client/server (C/S) web applications based on Java Sockets. In summary, when creating a Java Socket-based web application, you need to write an application to both the server side and the client side.

Keywords

Java Socket; wireless LAN; chat software.

1. Introduction

With the rapid development of Internet, the development demand of network application software is increasing day by day. There is an urgent need for a variety of network application software, but also spawned a variety of network programming technology. Using socket to achieve process communication on the Internet, and then to achieve a variety of network application functions, is an important way to develop network applications. We are familiar with network programs, such as QQ, MSN and so on. These network applications all use socket related technologies. It can be seen that socket plays an important role in network application communication.

1.1. Java Socket Overview

The original socket programming interface is Berkeley socket under UNIX operating system, which is very complicated. In order to facilitate network programming, Java platform provides us with a set of powerful classes, which can be used to develop network programs. Java is an object-oriented programming development language. It not only absorbs the advantages of C + + language, but also discards the concepts of multi inheritance and pointer that are difficult to understand in C + +. Therefore, Java language has the remarkable characteristics of security, cross platform, object-oriented, simple, and applicable to the network.

1.1.1 socket classification

Users can use two kinds of sockets: streaming socket and datagram socket. The streaming socket uses the transmission control protocol (TCP), and the datagram socket uses the user datagram protocol (UDP).

Streaming socket provides reliable connection oriented streaming service, similar to telephone system. Every complete data transmission must go through the process of establishing connection, using connection and terminating connection. In the process of data transmission,

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

each data packet does not carry the destination address, but uses the connection ID. TCP protocol provides the connection oriented virtual circuit. In essence, the connection is a pipeline. The data flows in from one end of the pipeline and out from the other end, not only in the same order, but also in the same content. In view of this, streaming socket has been more and more widely used, usually used for data file transmission, such as FTP, Telnet, etc., especially suitable for the transmission of large quantities, orderly, non repetitive data.

Datagram socket provides connectionless datagram transmission service and supports two-way data flow. Data is transmitted through independent datagrams, similar to postal system. Each packet carries a complete destination address, and each packet is transmitted independently in the system. However, connectionless service can not guarantee the sequence of packets, do not recover and retransmit the wrong packets, and do not guarantee the reliability of transmission. An important feature of datagram socket is the record boundary it keeps. For this feature, datagram socket adopts a very similar model to many packet switching networks. Datagram socket mode can achieve high communication speed because it cancels the retransmission verification mechanism. It can be used as some aspects of communication that do not require high data reliability, such as real-time voice, image forwarding and broadcast messages. Although most chat software use UDP protocol to ensure real-time, but I use TCP protocol to ensure the accuracy of chat content.

1.1.2 java socket communication mechanism

Java socket, also known as "socket", is used to describe the IP address and port. It is the handle of a communication chain Two Android programs on the network exchange data through a two-way communication connection. One end of the two-way link is a socket. Socket is usually used to realize the connection between client and server. It is mainly determined by an IP address and a port number. In Android development, java socket programming mainly refers to network programming based on TCP / IP protocol.

Android applications use the java socket API (Application Programming Interface) to realize the communication between each other. Each socket is associated with a process. It also uses the lower layer network communication protocol functions and operating system calls to achieve the actual communication work. In fact, socket is only an abstraction. It is a two-way endpoint of connection, through which data can be sent or received.

2. Wifi Implementation Principle

The chat software introduced in this paper is based on WiFi communication. Next, the main content of WiFi communication will be introduced. It is worth noting that I only introduce a simple point-to-point method which can carry out WiFi communication, and WiFi connection programming is not limited to this one.

Because Android has many interfaces that can call the underlying application, such as the permission to open WiFi communication, access to the IP address of the local machine under the current network and other information, it is very convenient to program, and can be called directly when necessary.

For the server, first create a WiFi hotspot, and then wait for the client to connect to the current LAN. If there is a socket connection request, data communication can be carried out after checking whether it is a legal connection. For the client, it needs to connect to the LAN created by the server first. After the connection is successful, it needs to find the IP address of the server so that the socket connection can be created later: one method is to manually configure the IP address of the server when the client is programming; the other method is to traverse all available IP addresses in the current network and establish the socket If the connection can be successfully established, the corresponding IP address of the current socket is the IP address of the server. The port number can be customized. Of course, if you want to realize WiFi

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

communication, you need some other details, including how to verify whether the corresponding network permissions are met, whether the server-side devices turn on the hotspot, how the client automatically connects to the LAN created by the server, and so on. The above content only provides a general idea.

3. Software Implementation

3.1. Main Code of Client

The following is the code for socket connection of the client. The principle is the same as that of most sockets. First, determine the IP address of the server and the port number monitored by the server, and then create a new socket through new mode. If the configuration is wrong, an error prompt can be thrown through toast.

```
public int Port = 10008;
public String ip = "192.168.43.30";
public void Connect() {
    try {
      socket = new Socket(ip,Port);
      if (socket.isConnected()){
        Looper.prepare();
        Toast.makeText(this,"success!",Toast.LENGTH_
                SHORT).show();
        Looper.loop();
      }
    } catch (IOException e) {
      Looper.prepare();
      Toast.makeText(this,”error”,Toast.LENGTH_SHORT).show();
      Looper.loop();
    }
}
```

Receivedata() is a data receiving and listening function. It is used by the client to receive data from the server. The specific execution process is as follows: first, start a new thread, then get the input stream of socket to the local input stream object, store the data in the local input stream with data in the buffer, and assign the length of the data stream to the variable bytes. Repeat the cycle. Once bytes > 0, that is, the data is stored in the input stream of the client, it will be read and stored in the string type variable STR, and then the UI in the main thread will be refreshed through the message method in the handler mechanism. The user interface can see the message from the server.

```
public void ReceiveData(){
  new Thread(){
    @Override
    public void run(){
     try {
      InputStream inputStream = socket.getInputStream();
      int bytes;
      byte[] buffer = new byte[4096];
      while (true) {
```

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

```
            bytes = inputStream.read(buffer);
            if (bytes > 0) {
              final byte[] data = new byte[bytes];
              System.arraycopy(buffer, 0, data, 0, bytes);
              String str = fromBytes(data);
              Message msg =new Message();
              msg.obj = str;
              mHandler.sendMessage(msg);
            }
          }
        } catch (IOException e) {e.printStackTrace();}
      }
    }.start();
      }
  public String fromBytes(byte[] bytes) {
     String str = new String(bytes, StandardCharsets.UTF_8);return str;
      }
  @SuppressLint("HandlerLeak")
  Handler mHandler = new Handler() {
      @Override
      public void handleMessage(Message msg) {
        super.handleMessage(msg);
        switch (msg.what) {
          case 0:
           String data = (String)msg.obj;
           bs.add(new ItemB( Main2Activity.this.getResources().
                          getDrawable(R.drawable.pic2), data, 0));
           lv.setAdapter(new MyAdapter());
            }
      }
  };
```

After clicking the send button, firstly judge the content in the dialog box. If it is empty, a Toast will pop up to prompt the user that the chat content is not allowed to be empty. Otherwise, the content is considered to be valid. First, use the getOutputStream method of the socket to get an output stream. The output stream obtained by the getOutputStream method on the Socket object of the client is actually the number sent to the server According to, the data in the edit box is sent to the server through the out. Writeutf () method.

```
send.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
       String string = sendContent.getText().toString();
       if (string.equals("")){
          Toast.makeText(Main2Activity.this,"no null ",Toast.LENGTH_SHORT).show();
       }else{
           try {
```

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

```
                DataOutputStream out = new DataOutputStream(socket.getOutputStream());
                out.writeUTF(string);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
});
```

3.2. Main Code of Server Side

Because the client code part has a detailed description of socket programming, and the server side is similar to it, so the repeated part will not be described, just for different parts.

```
try {
    server = new ServerSocket(PORT);
    mExecutorService = Executors.newCachedThreadPool();
    System.out.println("server start");
    Socket client = null;
    while(true) {
        client = server.accept();
        mList.add(client);
        mExecutorService.execute(new Service(client));
    }
  }catch (Exception e) {
        e.printStackTrace();
  }
  class Service implements Runnable {
      private Socket socket;
      private BufferedReader in = null;
      private String msg = "";
      public Service(Socket socket) {
          this.socket = socket;
          try {
            in = new BufferedReader(new InputStreamReader(
                                        socket.getInputStream()));
             msg = "server address:" +this.socket.getInetAddress() +
                                    "come toal:"+mList.size()+"(server send)";
             this.sendmsg();
          } catch (IOException e) {
              e.printStackTrace();
          }
      }
    public void run() {
       try {
          while(true) {
            if((msg = in.readLine())!= null) {
```

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

```
            if(msg.equals("exit")) {
               mList.remove(socket);
               in.close();
               msg = "user:" + socket.getInetAddress()
                    + "exit total:" + mList.size();
               socket.close();
              this.sendmsg();
              break;
          } else {
            msg = socket.getInetAddress() + ":" + msg;
            this.sendmsg();
           }
        }
      }catch (Exception e) {e.printStackTrace();}
   }
  public void sendmsg() {
     System.out.println(msg);
     int num = mList.size();
     for (int index = 0; index < num; index ++) {
          Socket mSocket = mList.get(index);
          PrintWriter pout = null;
          try {pout = new PrintWriter(new BufferedWriter(new
                              OutputStreamWriter(mSocket.getOutputStream())),true);
              pout.println(msg);
            }catch (IOException e) {e.printStackTrace();}
        }
      }
     }
```

4. Concluding Remarks

Using java socket to develop network application program can shield the complex structure and protocol of network. Therefore, the developed application software can run on all kinds of networks, and it is very easy to realize the interconnection of heterogeneous networks without caring about the specific network, the specific location of servers or clients on the network and the details of data transmission. And compared with using winsocket to write network applications, javasocket is more simple and convenient, the program structure is clear, and it is easy to upgrade and maintain. Although the scope of application of this software has some limitations, it is of great significance to understand and understand java socket programming. As this article focuses on socket programming, there is no detailed introduction to other contents of Android programming.

Acknowledgments

This paper was financially supported by “the Fundamental Research Funds for the Central Universities(2016MS122).

ISSN: 2637-6067 DOI: 10.6918/IJOSSER.202006_3(6).0009

References

[1] Li Zhixin, Yang Ruilong. Network programming in Java programming [M]. Beijing: Tsinghua University Press, 2009.

[2] Ma Xiaomin, Xiao Ming, Jiang Yuanming, et al. Java network programming principle and JSP Web development core technology [M]. Beijing: China Railway Press, 2010.

[3] Yao Xiaofang, Shu Xiaosong. Basic application research based on java socket network programming [J]. Wireless Internet technology, 2017 (22): 32-33.

[4] Zhang Yungang, Liu Changchun, Liu Wei, he Fuzhi. Remote monitoring system based on socket and multithreading [J]. Control engineering, 2006,02:175-177.

[5] Li Xianfan, Gao Jianrong. Development and implementation of LAN instant messaging system based on socket programming interface [J]. Journal of Changzhou Light Industry Vocational and technical college, 2006,04:14-18.