

华东师范大学数据科学与工程学院实验报告

课程名称：计算机网络与编程

年级：2020 级

上机实践成绩：

指导教师：张召

姓名：张熙翔

学号：10205501427

上机实践名称：基于 TCP 的 Socket 编程

上机实践日期：2022/4/24

一、实验目的

学习使用 `ServerSocket` 和 `Socket` 实现 TCP 通信
了解粘包的概念

二、实验任务

使用 `ServerSocket` 和 `Socket` 编写代码
解决粘包问题

三、使用环境

IntelliJ IDEA 2020.3.2
JDK 11.0.6

四、实验过程

Task1:分别运行修改后的 `Server` 端和 `Client` 端代码，并将运行结果附在实验报告中。

Server 端：

```
阻塞等待客户端连接中...
我是服务器，客户端说： 用户名：ECNUDase； 密码：123
我是服务器，已经回复了客户端消息

进程已结束,退出代码0
```

Client 端：

```
我是客户端，服务器说： 欢迎您！

进程已结束,退出代码0
```

Task2:使用 `Scanner` 修改 `TCPClient` 类，同时修改服务器代码，达成如下效果，请将实验结果附在实验报告中。

客户端读取用户输入的一行字符并将数据发送给服务器

服务器将收到的字符全部转换为大写字母，服务器将修改后的数据发送给客户端

客户端收到修改后的数据，并在其屏幕上显示

服务器：

阻塞等待客户端连接中...

我是服务器，收到客户端字符： zhangxixiang

我是服务器，已经转换为大写，并回复客户端

进程已结束,退出代码0

```
1. import java.io.*;
2. import java.net.InetSocketAddress;
3. import java.net.ServerSocket;
4. import java.net.Socket;
5. import java.nio.charset.StandardCharsets;
6. import java.util.ArrayList;
7.
8. public class TCPServer {
9.     private static final int PORT = 9091;
10.    public static void main(String[] args) throws IOException {
11.        ServerSocket serverSocket = new ServerSocket();
12.        serverSocket.setReuseAddress(true);
13.        serverSocket.setReceiveBufferSize(64 * 1024 * 1024);
14.        serverSocket.bind(new InetSocketAddress("127.0.0.1", PORT), 50);
15.
16.        System.out.println("阻塞等待客户端连接中...");
17.        Socket socket = serverSocket.accept();
18.
19.        InputStream is = socket.getInputStream();
20.        InputStreamReader isr = new InputStreamReader(is);
21.        BufferedReader br = new BufferedReader(isr);
22.        String info;
23.        ArrayList list = new ArrayList();
24.        while(true){
25.            info = br.readLine();
26.            if(info != null){
27.                list.add(info.toUpperCase());
28.                System.out.println("我是服务器，收到客户端字符： " + info);
29.            }else {
30.                break;
31.            }
32.        }
33.        list.add(null);
34.        socket.shutdownInput();
```

```
35.
36.     OutputStream os = socket.getOutputStream();
37.     int i=0;
38.     while(list.get(i)!=null){
39.         String message = list.get(i).toString();
40.         i++;
41.         os.write(message.getBytes(StandardCharsets.UTF_8));
42.     }
43.
44.     System.out.println("我是服务器，已经转换为大写，并回复客户端");
45.     socket.shutdownOutput();
46.
47.     os.close();
48.     br.close();
49.     isr.close();
50.     is.close();
51.     socket.close();
52.     serverSocket.close();
53. }
54. }
```

客户端：

请输入一行字符：

zhangxixiang

我是客户端，服务器将字符转换为大写： ZHANGXIXIANG

进程已结束,退出代码0

```
1. import java.io.*;
2. import java.util.*;
3. import java.net.InetSocketAddress;
4. import java.net.Socket;
5. import java.nio.charset.StandardCharsets;
6. public class TCPClient {
7.     private static final int PORT = 9091;
8.     public static void main(String[] args) throws IOException {
9.
10.         Socket socket = new Socket();
11.         socket.connect(new InetSocketAddress("127.0.0.1", PORT),3000);
12.
13.         OutputStream os = socket.getOutputStream();
14.         //String message = "用户名: ECNUdase; 密码: 123";
15.         System.out.println("请输入一行字符: ");
16.
17.         Scanner in = new Scanner(System.in);
```

```
18.     String message= in.nextLine();
19.
20.     os.write(message.getBytes(StandardCharsets.UTF_8));
21.     socket.shutdownOutput();
22.
23.     InputStream is = socket.getInputStream();
24.     InputStreamReader isr = new InputStreamReader(is);
25.     BufferedReader br = new BufferedReader(isr);
26.     while(true){
27.         String info = br.readLine();
28.         if(info != null){
29.             System.out.println("我是客户端，服务器将字符转换为大写: " + info);
30.         }else {
31.             break;
32.         }
33.     }
34.     socket.shutdownInput();
35.
36.     br.close();
37.     isr.close();
38.     is.close();
39.     os.close();
40.     socket.close();
41. }}
```

Task3:在 Task2 基础上继续修改 TCPServer 类和 TCPClient 类，Client 端能够分别发送从控制台输入的多行数据，Server 端收到后分别发送回客户端（请测试数据为中文时的情况）。请将实验结果附在实验报告中。

服务器：

```
1.  import java.io.*;
2.  import java.net.InetSocketAddress;
3.  import java.net.ServerSocket;
4.  import java.net.Socket;
5.  import java.util.ArrayList;
6.
7.  public class TCPServer {
8.      private static final int PORT = 9091;
9.      public static void main(String[] args) throws IOException {
10.
11.         ServerSocket serverSocket = new ServerSocket();
12.         serverSocket.setReuseAddress(true);
13.         serverSocket.setReceiveBufferSize(64 * 1024 * 1024);
14.         serverSocket.bind(new InetSocketAddress("127.0.0.1", PORT), 50);
15.         while(true) {
16.             System.out.println("阻塞等待客户端连接中...");
17.             Socket socket = serverSocket.accept();
```

```
18.      InputStream is = socket.getInputStream();
19.      InputStreamReader isr = new InputStreamReader(is);
20.      BufferedReader br = new BufferedReader(isr);
21.      OutputStream os = socket.getOutputStream();
22.      PrintWriter printWriter = new PrintWriter(os);
23.      String info;
24.      ArrayList list = new ArrayList();
25.      while (true) {
26.          info = br.readLine();
27.          if (info != null) {
28.              list.add(info.toUpperCase());
29.              System.out.println("我是服务器，收到客户端字符: " + info);
30.          } else {
31.              break;
32.          }
33.      }
34.      list.add(null);
35.      socket.shutdownInput();
36.      int i = 0;
37.      while (list.get(i) != null) {
38.          String message = list.get(i).toString();
39.          i++;
40.          printWriter.println(message);
41.          printWriter.flush();
42.      }
43.      System.out.println("我是服务器，已经转换为大写，并回复客户端");
44.      socket.shutdownOutput();
45.  }
46.  }
47. }
```

阻塞等待客户端连接中...

我是服务器，收到客户端字符: 张熙翔

我是服务器，已经转换为大写，并回复客户端

阻塞等待客户端连接中...

我是服务器，收到客户端字符: is handsome.

我是服务器，已经转换为大写，并回复客户端

阻塞等待客户端连接中...

客户端：

```
1. import java.io.*;
2. import java.util.*;
3. import java.net.InetSocketAddress;
4. import java.net.Socket;
5. public class TCPClient {
6.     private static final int PORT = 9091;
7.     public static void main(String[] args) throws IOException {
8.         while(true) {
9.             Socket socket = new Socket();
10.            socket.connect(new InetSocketAddress("127.0.0.1", PORT),3000);
11.            OutputStream os = socket.getOutputStream();
12.            PrintWriter printWriter = new PrintWriter(os);
13.            System.out.println("请输入一行字符: ");
14.            Scanner in = new Scanner(System.in);
15.            String message = in.nextLine();
16.            printWriter.println(message);
17.            printWriter.flush();
18.            socket.shutdownOutput();
19.            InputStream is = socket.getInputStream();
20.            InputStreamReader isr = new InputStreamReader(is);
21.            BufferedReader br = new BufferedReader(isr);
22.            while (true) {
23.                String info = br.readLine();
24.                if (info != null) {
25.                    System.out.println("我是客户端，服务器将字符转换为大写: " + info);
26.                } else {
27.                    break;
28.                }
29.            }
30.            socket.shutdownInput();
31.        }
32.    } }
```

请输入一行字符：

张熙翔

我是客户端，服务器将字符转换为大写： 张熙翔

请输入一行字符：

is handsome.

我是客户端，服务器将字符转换为大写： IS HANDSOME.

请输入一行字符：

Task4:查阅资料，总结半包粘包产生的原因以及相关解决方案，并附在实验报告中。

TCP 传输中，客户端发送数据，实际是把数据写入到了 TCP 的缓存中，粘包和半包也会在此时产生。

如果客户端发送的包的大小比 TCP 的缓存容量小，并且 TCP 缓存可以存放多个包，那么客户端和服务端的一次通信就可能传递了多个包，这时候服务端从 TCP 缓存就可能一下读取了多个包，这种现象就叫粘包。

如果客户端发送的包的大小比 TCP 的缓存容量大，那么这个数据包就会被分成多个包，通过 Socket 多次发送到服务端，服务端第一次从接受缓存里面获取的数据，实际是整个包的一部分，这时候就产生了半包。

粘包的主要原因:

发送方每次写入数据 < 套接字(Socket)缓冲区大小

接收方读取套接字(Socket)缓冲区数据不够及时

半包的主要原因:

发送方每次写入数据 > 套接字(Socket)缓冲区大小

发送的数据大于协议的最大传输单元，因此必须拆包

解决方法:

1. 将 TCP 连接改成短连接，一个请求一个短连接。使得建立连接到释放连接之间的消息即为传输的信息，消息也就产生了边界。

2. 封装成帧，也就是原本发送消息的单位是缓冲大小，现在换成了帧，可以自定义边界了。

3. 固定长度，消息边界也就是固定长度。

4. 分隔符号，消息边界也就是分隔符本身。

5. 一个专门的字段存储消息的长度。作为服务端，接受消息时，先解析固定长度的字段获取消息总长度，然后读取后续内容。