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：**2021	**上机实践成绩：**
指导教师：张召	**姓名：温兆和**	**学号：**10205501432
上机实践名称：Socket 编程优化		**上机实践日期：**2022.05.13
上机实践编号：12	**组号：**001-432	**上机实践时间：**13：00

一、 实验目的

实现单服务端多客户端通信；
对数据发送和接受进行优化。

二、 实验任务

实现单服务端多客户端的 TCP Socket 通信；
将数据发送与接受并行。

三、 使用环境

Intellij IDEA 2020.3.2
JDK 11.0.6

四、 实验过程

Task1. 分别启动一个 TCP Server 和多个 TCP Client，将运行结果附在实验报告中。

结果：

"C:\jdk-18_windows-x64_bin (1)\jdk-18.0.1.1\bin\java.exe" "-javaagent:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\lib\idea_rt.jar=52127:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\HUAWEI\IdeaProjects\untitled2\out\production\untitled2 WEIZHI.TCPServer
Server hears127.0.0.1:9091
Blocked and waiting for the client connect...
Client number: 1
Blocked and waiting for the client connect...
New Client connect127.0.0.1:52134
Receive client messagesUser: Dase; Code:ecnudase
Client number: 2
Blocked and waiting for the client connect...
New Client connect127.0.0.1:52140
Receive client messagesUser: Dase; Code:ecnudase
Client number: 3
Blocked and waiting for the client connect...
New Client connect127.0.0.1:52146
Receive client messagesUser: Dase; Code:ecnudase
Client number: 4
Blocked and waiting for the client connect...
New Client connect127.0.0.1:52151
Receive client messagesUser: Dase; Code:ecnudase

Process finished with exit code 130

Task2. 修改 Client 类，使其发送和接受并行，即当和服务器端连接时，可随时发送和接

收信息，将修改后的 Client 代码附在实验报告中。

代码：

客户端：

```
package WEIZHI;
import java.net.InetSocketAddress;
import java.io.*;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;
public class TCPClient {
    private final int port;
    private ServerListener serverListener;
    private List<ServerHandler> serverHandlerList = new ArrayList<>();
    private List<ServerWriteHandler> serverWriteHandlerList = new ArrayList<>();
    public TCPClient(int port){
        this.port = port;
    }
    public void start() throws IOException {
        serverListener = new ServerListener(port);
        serverListener.start();
    }

    private class ServerListener extends Thread {
        private final int port;
        ServerListener(int port) throws IOException {
            this.port=port;
        }

        @Override
        public void run(){
            Socket socket = new Socket();
            try {
                socket.connect(new InetSocketAddress("127.0.0.1", port),3000);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            try{
                ServerHandler serverHandler = new ServerHandler(socket);
                serverHandler.start();
                serverHandlerList.add(serverHandler);
            }catch(IOException e) {
                e.printStackTrace();
                System.out.println("Wrong server connect..." + e.getMessage());
            }
            }}

}

class ServerWriteHandler extends Thread {
        InputStream in = System.in;
        BufferedReader input = new BufferedReader(new InputStreamReader(in));
        private final OutputStream outputStream;

    ServerWriteHandler(OutputStream outputStream) {
        this.outputStream = outputStream;
    }
    @Override
    public void run() {
        PrintStream socketPrintstream = new PrintStream(outputStream);
        String str;
```

```
        do {
            try {
                str = this.input.readLine();
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            socketPrintstream.println(str);
        } while (str!=null);
    }
}

class ServerReadHandler extends Thread {
    private final InputStream inputStream;
    ServerReadHandler(InputStream inputStream){
        this.inputStream = inputStream;
    }
    @Override
    public void run(){
        try{
            BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
            while(true){
                String str = bufferedReader.readLine();
                if(str == null){
                    System.out.println("No data found");
                    break;
                }else{
                    System.out.println("Data got: " + str);
                }
            }
        }catch (IOException e){
            e.printStackTrace();
        }}
}
class ServerHandler extends Thread {
    private Socket socket;
    private final ServerReadHandler serverReadHandler;
    private final ServerWriteHandler serverWriteHandler;
    ServerHandler(Socket socket) throws IOException{
        this.socket = socket;
        this.serverReadHandler = new ServerReadHandler(socket.getInputStream());
        this.serverWriteHandler = new ServerWriteHandler(socket.getOutputStream());
    }
    @Override
    public void run() {
        super.run();
        serverWriteHandler.start();
        serverReadHandler.start();
    }

}

class TestClient {
    private static final int PORT = 9091;

    public static void main(String[] args) throws IOException {
        TCPClient client = new TCPClient(PORT);
        client.start();
    }
}
```

服务器端：

```
package WEIZHI;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.Scanner;
import java.util.List;
public class TCPServer {
    private final int port;
    private ClientListener clientListener;
    private List<ClientHandler> clientHandlerList = new ArrayList<>();
    public TCPServer(int port){
        this.port = port;
    }
    public void start() throws IOException {
        clientListener = new ClientListener(port);
        clientListener.start();
    }
    public void broadcast(String str) {
        for(ClientHandler clientHandler: clientHandlerList){
            clientHandler.send(str);
        }
    }
    private class ClientListener extends Thread {
        private ServerSocket serverSocket;
        ClientListener(int port) throws IOException {
            serverSocket = new ServerSocket(port);
        }

        @Override
        public void run(){
            while(true){
                try{
                    System.out.println("Blocked and waiting for the client
connect...");
                    Socket socket = serverSocket.accept();
                    ClientHandler clientHandler = new ClientHandler(socket);
                    clientHandler.start();
                    clientHandlerList.add(clientHandler);
                }catch(IOException e) {
                    e.printStackTrace();
                    System.out.println("Wrong client connect..." + e.getMessage());
                }
            }}}
}
class ClientReadHandler extends Thread {
    private final InputStream inputStream;
    ClientReadHandler(InputStream inputStream){
        this.inputStream = inputStream;
    }
    @Override
    public void run(){
        try{
            BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
            while(true){
                String str = bufferedReader.readLine();
                if(str == null){
                    System.out.println("No data found");
```

```
                break;
            }else{
                System.out.println("Data got: " + str);
            }
        }
    }catch (IOException e){
        e.printStackTrace();
    }}
}
class ClientWriteHandler extends Thread {
    private final PrintStream printStream;
    private final ExecutorService threadPool;
    ClientWriteHandler(OutputStream outputStream) {
        this.printStream = new PrintStream(outputStream);
        this.threadPool = Executors.newSingleThreadExecutor();
    }
    void send(String str){
        this.threadPool.execute(new WriteRunnable(str));
    }
    class WriteRunnable implements Runnable {
        private final String msg;
        WriteRunnable(String msg){
            this.msg = msg;
        }
        @Override
        public void run() {
            ClientWriteHandler.this.printStream.println(msg);
        }
    }
}
class ClientHandler extends Thread {
    private Socket socket;
    private final ClientReadHandler clientReadHandler;
    private final ClientWriteHandler clientWriteHandler;
    ClientHandler(Socket socket) throws IOException{
        this.socket = socket;
        this.clientReadHandler = new
                ClientReadHandler(socket.getInputStream());
        this.clientWriteHandler = new
                ClientWriteHandler(socket.getOutputStream());
    }
    @Override
    public void run() {
        super.run();
        clientReadHandler.start();
        clientWriteHandler.start();
    }
    public void send(String str) {
        clientWriteHandler.send(str);
    }
}
class TestServer {
    private static final int PORT = 9091;

    public static void main(String[] args) throws IOException {
        TCPServer server = new TCPServer(PORT);
        server.start();
        Scanner scanner = new Scanner(System.in);
        while (scanner.hasNext()) {
            String s = scanner.next();
            server.broadcast(s);
```

```
        }
    }
}
```

结果：

客户端：

"C:\jdk-18_windows-x64_bin (1)\jdk-18.0.1.1\bin\java.exe" "-javaagent:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\lib\idea_rt.jar=52795:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\HUAWEI\IdeaProjects\untitled2\out\production\untitled2 WEIZHI.TestClient

bgtrhbt
renrjh
Data got: ergntjhnt
Data got: ergjr
gtnhtjhr
Data got: rghtgth
egjththth
Data got: efrnjghthr
jkhtjhyhjynh

Process finished with exit code 130

服务器端：

"C:\jdk-18_windows-x64_bin (1)\jdk-18.0.1.1\bin\java.exe" "-javaagent:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\lib\idea_rt.jar=52790:C:\Program Files (x86)\Windows Mail\IntelliJ IDEA 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\HUAWEI\IdeaProjects\untitled2\out\production\untitled2 WEIZHI.TestServer

Blocked and waiting for the client connect...
Blocked and waiting for the client connect...
Data got: bgtrhbt
Data got: renrjh
ergntjhnt
ergjr
Data got: gtnhtjhr
rghtgth
Data got: egjththth
efrnjghthr
Data got: jkhtjhyhjynh
java.net.SocketException: Connection reset
    at java.base/sun.nio.ch.NioSocketImpl.implRead(NioSocketImpl.java:320)
    at java.base/sun.nio.ch.NioSocketImpl.read(NioSocketImpl.java:347)
    at java.base/sun.nio.ch.NioSocketImpl$1.read(NioSocketImpl.java:800)
    at java.base/java.net.Socket$SocketInputStream.read(Socket.java:966)
    at java.base/sun.nio.cs.StreamDecoder.readBytes(StreamDecoder.java:270)
    at java.base/sun.nio.cs.StreamDecoder.implRead(StreamDecoder.java:313)
    at java.base/sun.nio.cs.StreamDecoder.read(StreamDecoder.java:188)
    at java.base/java.io.InputStreamReader.read(InputStreamReader.java:176)
    at java.base/java.io.BufferedReader.fill(BufferedReader.java:162)
    at java.base/java.io.BufferedReader.readLine(BufferedReader.java:329)
    at java.base/java.io.BufferedReader.readLine(BufferedReader.java:396)
    at WEIZHI.ClientReadHandler.run(TCPServer.java:60)

Process finished with exit code 130

五、总结

在本周的实验中，我们实现单服务端多客户端通信并对数据发送和接受进行了优化，为以后的学习和实践打下了基础。