微光后端第十一题

编写一个聊天客户端

- 代码如下

```
public class SimpleChatClientA {
    private SocketChannel socketChannel;//定义一个socketchannel以完成连接与通道的建立

    public void go() {
        setUpNetworking();
        Scanner scanner = new Scanner(System.in);
        while (true) {
            Thread t = new Thread(new ClientHandler(socketChannel));//把接受信息改为另一线程
            t.start();
            String message = scanner.next();//主线程
            if ("退出".equals(message)) break;
            else sendMessage(message);
        }
    }

    private void setUpNetworking() {//连接客户端
        try {
            InetSocketAddress serverAddress = new InetSocketAddress("127.0.0.1", 5000);
            socketChannel = SocketChannel.open(serverAddress);
            System.out.println("已连接");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void sendMessage(String message) {
        if (socketChannel != null && socketChannel.isOpen()) {//运用printwriter实现信息的发送
            try {
                Writer writer = Channels.newWriter(socketChannel, StandardCharsets.UTF_8);
                PrintWriter printWriter = new PrintWriter(writer);
                printWriter.println(message);
                printWriter.flush();
                }catch (Exception e) {
                System.out.println("错误");
            }
        }
    }
```

```
    public static void main(String[] args) {//运行
        SimpleChatClientA simpleChatClientA = new SimpleChatClientA();
        simpleChatClientA.go();
     }

    public class ClientHandler implements Runnable {//定义了接受信息的方法，原理与服务端相同
        private SocketChannel socketChannel;

        public ClientHandler(SocketChannel socketChannel) {
            this.socketChannel = socketChannel;
        }

        @Override
        public void run() {//方法重写
            try {
                Reader reader = Channels.newReader(socketChannel, StandardCharsets.UTF_8);
                BufferedReader bufferedReader = new BufferedReader(reader);
                String message;
                while ((message = bufferedReader.readLine()) != null) {
                    System.out.println(message);;
                }
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        }
    }
}
```

编写一个聊天服务器

- 代码如下

```
ublic class SimpleChatServer {
    public static void main(String[] args) {
        new SimpleChatServer().go();
    }
    public void go() {
        try {
            ServerSocketChannel serverChannel = ServerSocketChannel.open();//打开通道
            serverChannel.bind(new InetSocketAddress(5000));//绑定端口
            while (true) {//循环接受连接
                SocketChannel clientChannel = serverChannel.accept();
                System.out.println("已连接");
```

```
            Thread t = new Thread(new ClientHandler(clientChannel));//多线程
            t.start();
            Thread t2 = new Thread(new Sent(clientChannel));
            t2.start();
        }
    } catch (IOException e) {
        System.out.println("错了");
    }
}

private void tellEveryone(String message) {
    System.out.println(message);//在服务器上输出信息
}


public class ClientHandler implements Runnable {
private SocketChannel socketChannel;

public ClientHandler(SocketChannel socketChannel) {
    this.socketChannel = socketChannel;//构造有参构造器，以便引入socketchannel
}

@Override
public void run() {//方法重写
    try {
        Reader reader = Channels.newReader(socketChannel, StandardCharsets.UTF_8);
        BufferedReader bufferedReader = new BufferedReader(reader);
        String message;
        while ((message = bufferedReader.readLine()) != null) {//当接受的信息不为0时
            tellEveryone(message);
        }
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
}

public class Sent implements Runnable {//此类用于输出信息
    private SocketChannel socketChannel;

    public Sent(SocketChannel socketChannel) {
        this.socketChannel = socketChannel;
    }

    @Override
```

```
        public void run() {
            Scanner scanner = new Scanner(System.in);
            try {
                Writer writer = Channels.newWriter(socketChannel, StandardCharsets.UTF_8);
                PrintWriter printWriter = new PrintWriter(writer);
                while (true) {
                    String message = scanner.nextLine();
                    if ("退出".equals(message)) {
                        break;
                    }
                    printWriter.println(message);
                    printWriter.flush();//刷新以保证输入的正确性
                }
            } catch (Exception e) {
                System.out.println("又错了");
            }
        }
    }
}
```

改进你的服务器

- 已包含在上述代码中
- 问题一：我们如何从服务器得到消息
- 让服务器把信息传回客户端，在客户端定义方法接受。（对于要给某个单独的客户端发送时，可以给客户端命名并在发送时调用）
- 问题二：我们应该何时从服务器得到消息
- 由于信息的时效性，自然是越快越好，此时可以建立多线程

PS：

- 因为快没时间了，所以写的有些简陋
- 如果答案实在有问题的话，可以给点提示和建议吗 qq：1187477643 多谢大佬（没时间也没关系）

参考网址：

Java网络编程（四）—— ServerSocket（一） java serversocket-CSDN博客

Java多线程（超详细！）-CSDN博客

[听好了，至此，一锤定音；尘埃，已然落定](项目——搭建一个web服务器 | 2024招新)

Java 多线程编程 | 菜鸟教程

已连接

asd

aaas

asd

asd

*asd*asd

asdsasddsaassdsaas

asaaqqqzsdsacsxsss

已连接

asd

aaas

asd

asd

asd

asdasdsasddsaassdsaas

asaaqqqzsdsacsxsss