华东师范大学数据科学与工程学院实验报告

课程名称： 分布式模型与编程	**年级：** 2016 级	**上机实践成绩：**
指导教师： 徐辰	**姓名：** 张宏伟	
上机实践名称： 预备知识	**学号：** 10165101180	**上机实践日期：** 20180914
上机实践编号：	**组号：**	**上机实践时间：**

一、 实验目的

掌握远程调用的概念，了解 Java Socket 编程和远程调用的实现。

二、 实验任务

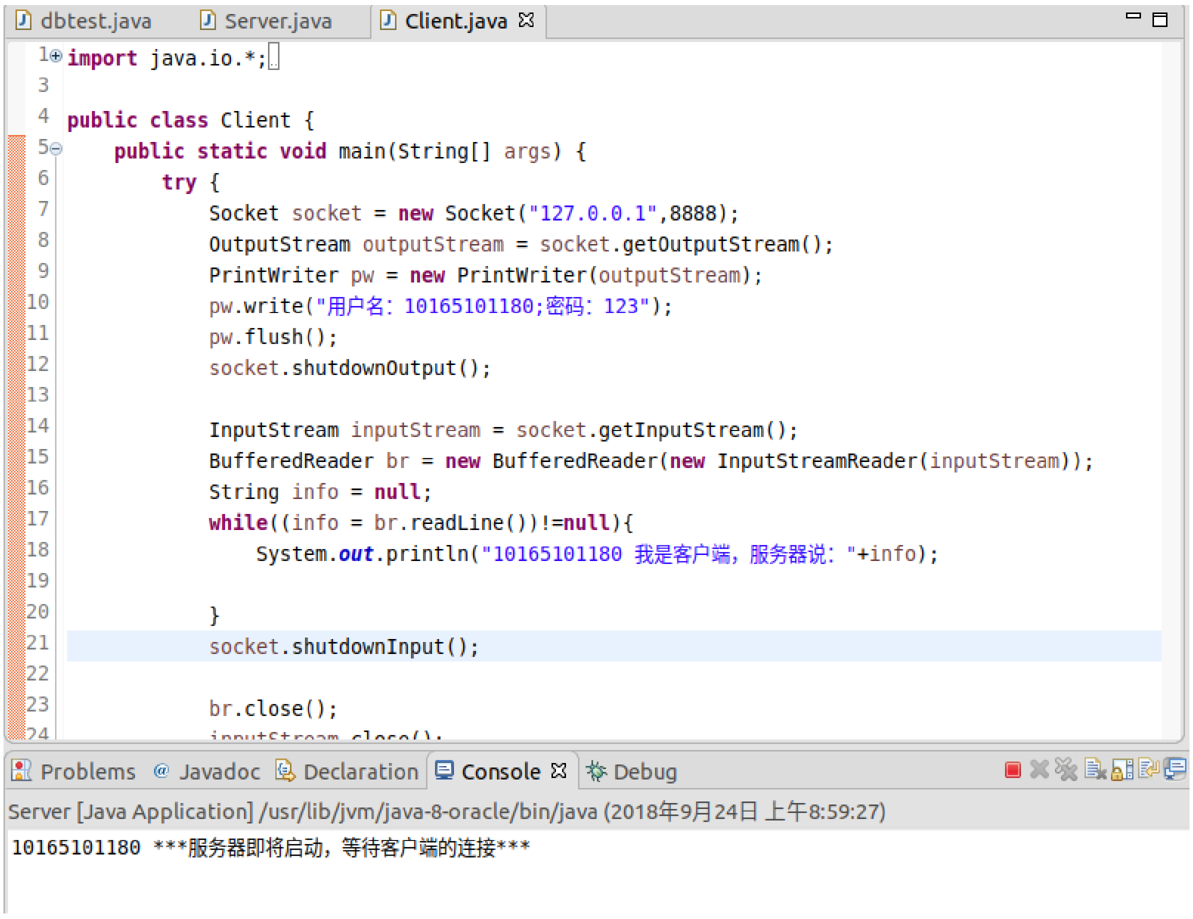
实现本地上客户端与服务器的通信与远程调用。

三、 使用环境

Ubuntu18.04+Eclipse

四、 实验过程

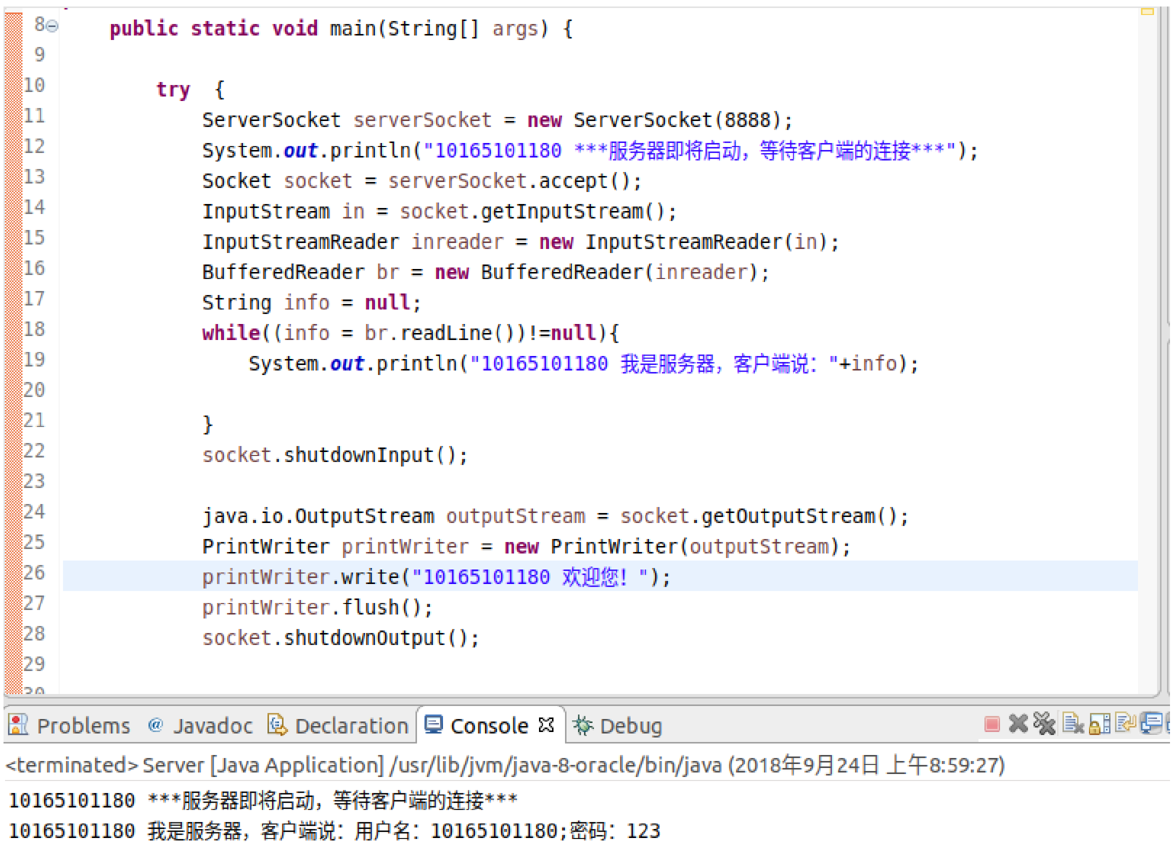
1.1 Scoket 编程-启动服务器



```java
import java.io.*;

public class Client {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("127.0.0.1",8888);
            OutputStream outputStream = socket.getOutputStream();
            PrintWriter pw = new PrintWriter(outputStream);
            pw.write("用户名：10165101180;密码：123");
            pw.flush();
            socket.shutdownOutput();

            InputStream inputStream = socket.getInputStream();
            BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
            String info = null;
            while((info = br.readLine())!=null){
                System.out.println("10165101180 我是客户端，服务器说："+info);

            }
            socket.shutdownInput();

            br.close();
```

```
Server [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午8:59:27)
10165101180 ***服务器即将启动，等待客户端的连接***
```

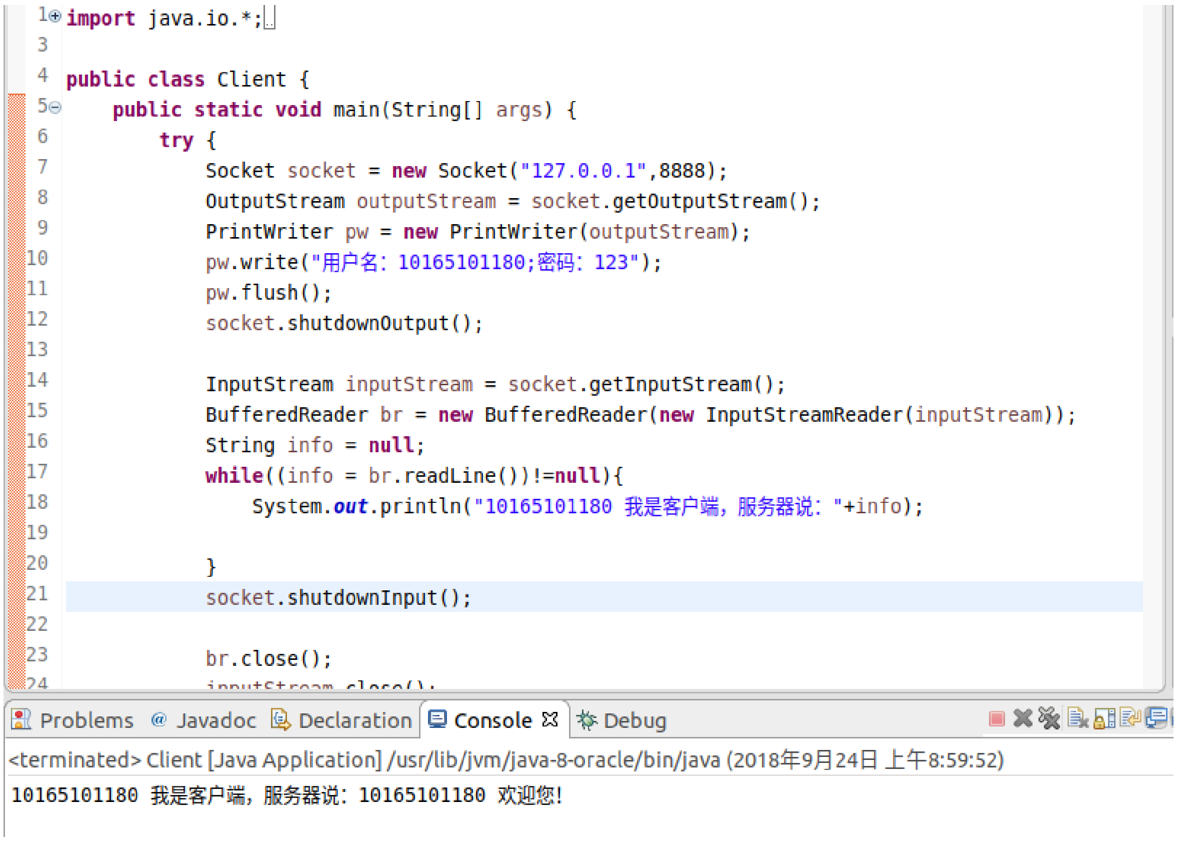
1.2 Scoket 编程-启动客户端，服务器端显示内容



```
    public static void main(String[] args) {

        try  {
            ServerSocket serverSocket = new ServerSocket(8888);
            System.out.println("10165101180 ***服务器即将启动，等待客户端的连接***");
            Socket socket = serverSocket.accept();
            InputStream in = socket.getInputStream();
            InputStreamReader inreader = new InputStreamReader(in);
            BufferedReader br = new BufferedReader(inreader);
            String info = null;
            while((info = br.readLine())!=null){
                System.out.println("10165101180 我是服务器，客户端说："+info);

            }
            socket.shutdownInput();

            java.io.OutputStream outputStream = socket.getOutputStream();
            PrintWriter printWriter = new PrintWriter(outputStream);
            printWriter.write("10165101180 欢迎您！");
            printWriter.flush();
            socket.shutdownOutput();
```

```
<terminated> Server [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午8:59:27)
10165101180 ***服务器即将启动，等待客户端的连接***
10165101180 我是服务器，客户端说：用户名：10165101180;密码：123
```

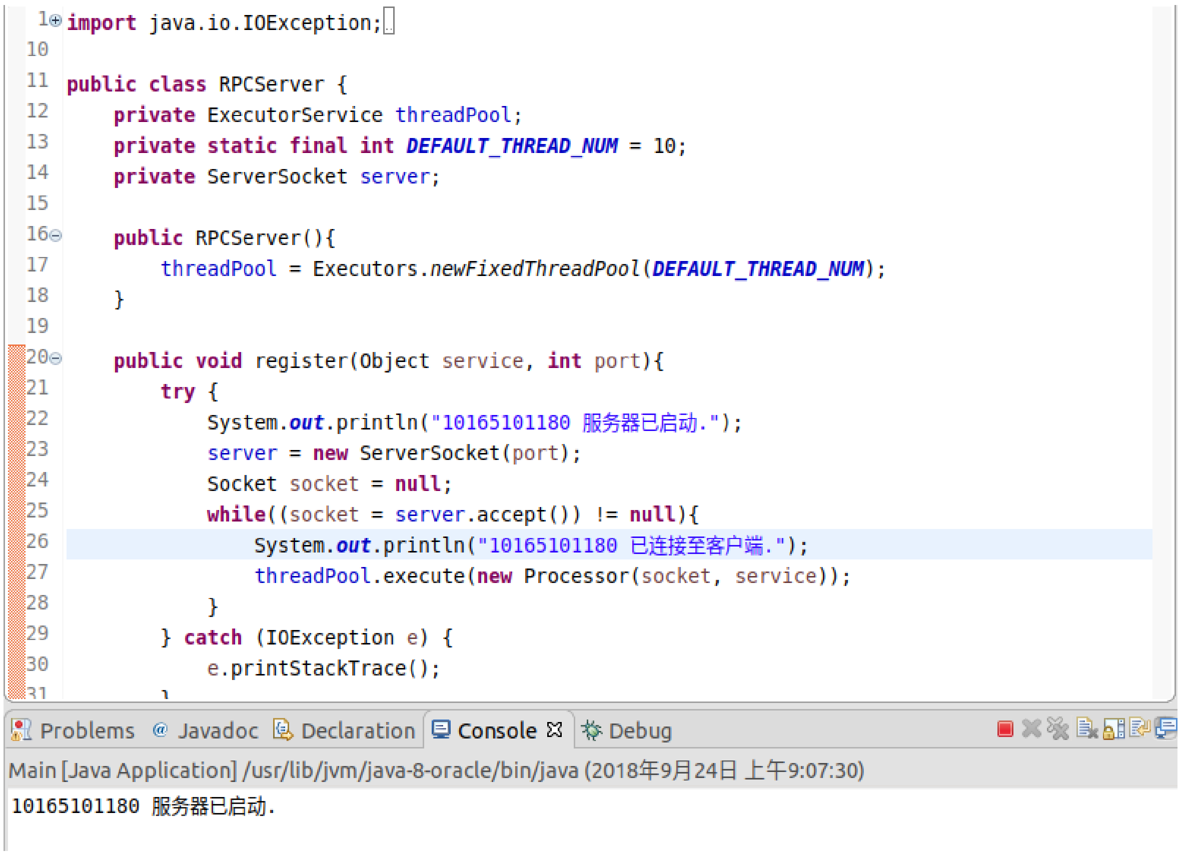
1.3 Socket 编程-启动客户端，客户端显示内容



```
import java.io.*;

public class Client {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("127.0.0.1",8888);
            OutputStream outputStream = socket.getOutputStream();
            PrintWriter pw = new PrintWriter(outputStream);
            pw.write("用户名：10165101180;密码：123");
            pw.flush();
            socket.shutdownOutput();

            InputStream inputStream = socket.getInputStream();
            BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
            String info = null;
            while((info = br.readLine())!=null){
                System.out.println("10165101180 我是客户端，服务器说："+info);

            }
            socket.shutdownInput();

            br.close();
            inputStream.close();
```

```
<terminated> Client [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午8:59:52)
10165101180 我是客户端，服务器说：10165101180 欢迎您！
```

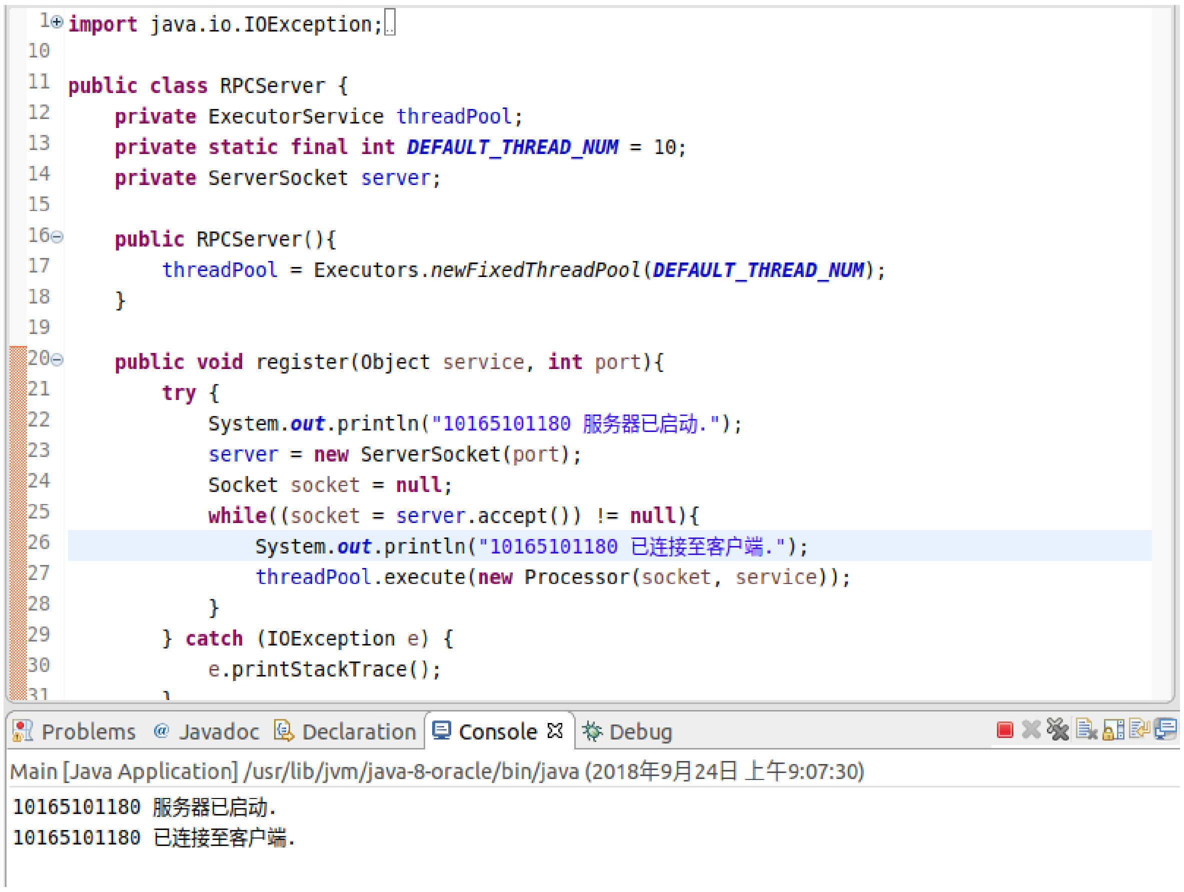
2.1 远程调用-启动服务器



```
import java.io.IOException;

public class RPCServer {
    private ExecutorService threadPool;
    private static final int DEFAULT_THREAD_NUM = 10;
    private ServerSocket server;

    public RPCServer(){
        threadPool = Executors.newFixedThreadPool(DEFAULT_THREAD_NUM);
    }

    public void register(Object service, int port){
        try {
            System.out.println("10165101180 服务器已启动.");
            server = new ServerSocket(port);
            Socket socket = null;
            while((socket = server.accept()) != null){
                System.out.println("10165101180 已连接至客户端.");
                threadPool.execute(new Processor(socket, service));
            }
        } catch (IOException e) {
            e.printStackTrace();
```

```
Main [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午9:07:30)
10165101180 服务器已启动.
```

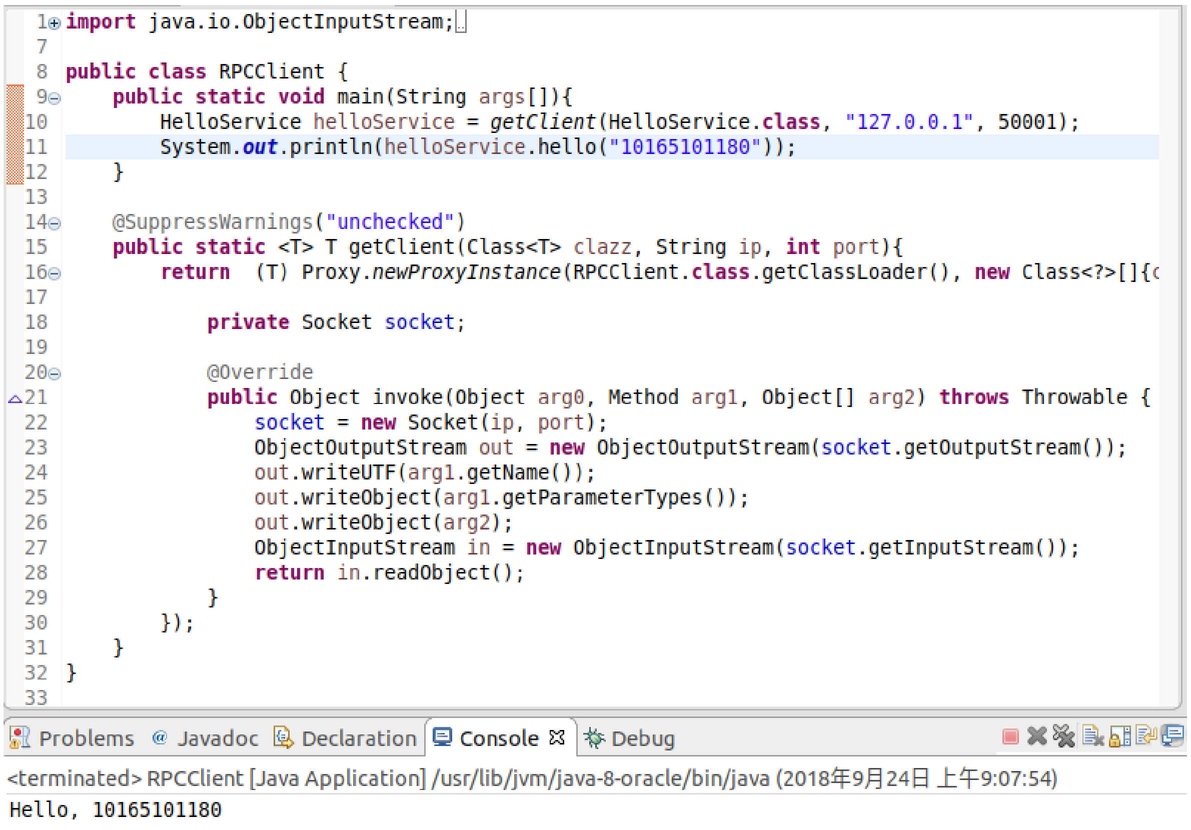
2.2 远程调用-启动客户端，服务器显示内容



```
import java.io.IOException;

public class RPCServer {
    private ExecutorService threadPool;
    private static final int DEFAULT_THREAD_NUM = 10;
    private ServerSocket server;

    public RPCServer(){
        threadPool = Executors.newFixedThreadPool(DEFAULT_THREAD_NUM);
    }

    public void register(Object service, int port){
        try {
            System.out.println("10165101180 服务器已启动.");
            server = new ServerSocket(port);
            Socket socket = null;
            while((socket = server.accept()) != null){
                System.out.println("10165101180 已连接至客户端.");
                threadPool.execute(new Processor(socket, service));
            }
        } catch (IOException e) {
            e.printStackTrace();
```

```
Main [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午9:07:30)
10165101180 服务器已启动.
10165101180 已连接至客户端.
```

2.3 远程调用-启动客户端，客户端显示内容



```
import java.io.ObjectInputStream;

public class RPCClient {
    public static void main(String args[]){
        HelloService helloService = getClient(HelloService.class, "127.0.0.1", 50001);
        System.out.println(helloService.hello("10165101180"));
    }

    @SuppressWarnings("unchecked")
    public static <T> T getClient(Class<T> clazz, String ip, int port){
        return  (T) Proxy.newProxyInstance(RPCClient.class.getClassLoader(), new Class<?>[]{c

                private Socket socket;

                @Override
                public Object invoke(Object arg0, Method arg1, Object[] arg2) throws Throwable {
                    socket = new Socket(ip, port);
                    ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());
                    out.writeUTF(arg1.getName());
                    out.writeObject(arg1.getParameterTypes());
                    out.writeObject(arg2);
                    ObjectInputStream in = new ObjectInputStream(socket.getInputStream());
                    return in.readObject();
                }
            });
    }
}
```

```
<terminated> RPCClient [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (2018年9月24日 上午9:07:54)
Hello, 10165101180
```

五、 **总结**

1.Socket 编程除了设置 ip 和端口外，操作与 IO 一致。

2.远程调用使用中，客户端和服务器都拥有调用函数的接口。客户端向服务器发送调用函数名与参数列表，服务器调用对应函数后返回给客户端结果。