

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

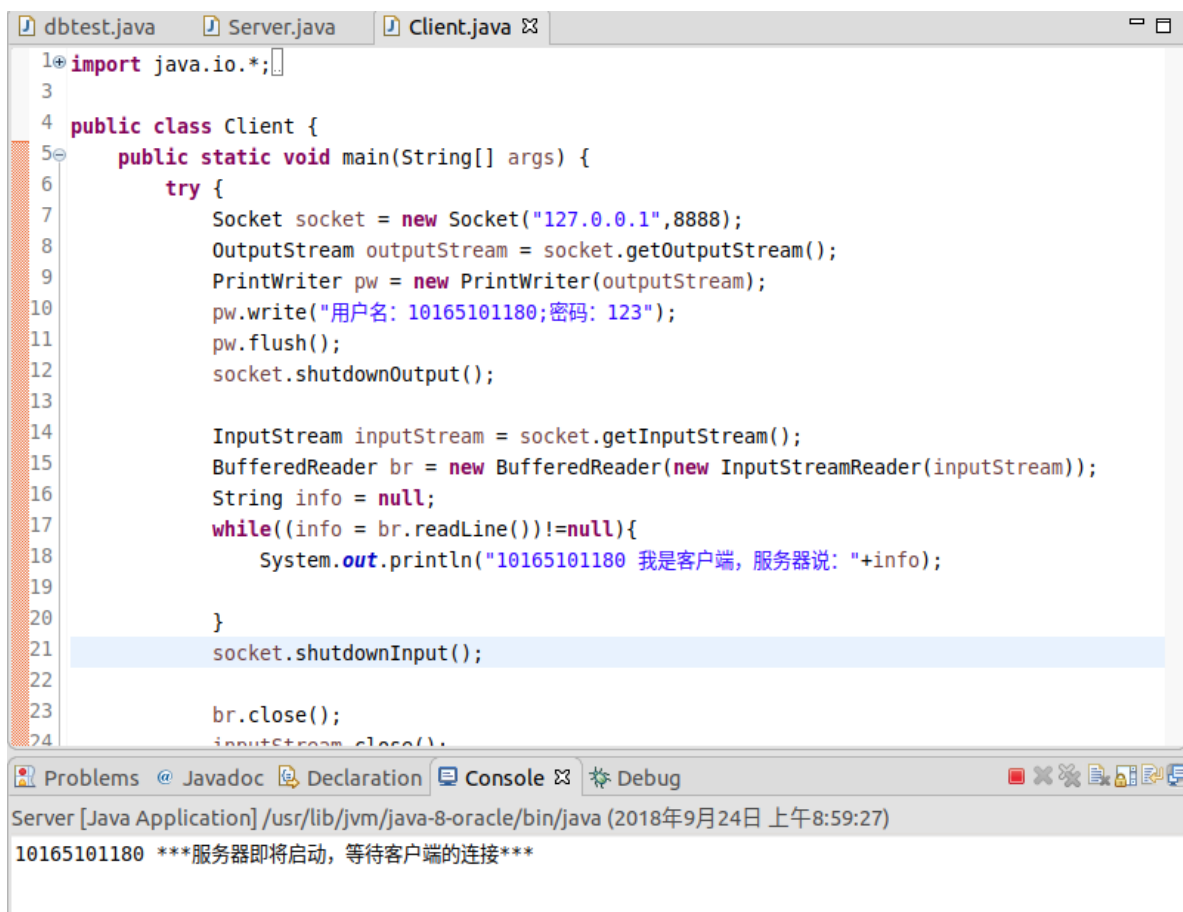
实现本地客户端与服务器的通信与远程调用。

三、使用环境

Ubuntu18.04+Eclipse

四、实验过程

1.1 Socket 编程-启动服务器



The screenshot shows the Eclipse IDE with three tabs: dbtest.java, Server.java, and Client.java. The Client.java file is active, displaying the following code:

```
1 import java.io.*;
3
4 public class Client {
5     public static void main(String[] args) {
6         try {
7             Socket socket = new Socket("127.0.0.1", 8888);
8             OutputStream outputStream = socket.getOutputStream();
9             PrintWriter pw = new PrintWriter(outputStream);
10            pw.write("用户名: 10165101180; 密码: 123");
11            pw.flush();
12            socket.shutdownOutput();
13
14            InputStream inputStream = socket.getInputStream();
15            BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
16            String info = null;
17            while((info = br.readLine()) != null){
18                System.out.println("10165101180 我是客户端, 服务器说: " + info);
19            }
20            socket.shutdownInput();
21
22            br.close();
23            inputStream.close();
24        } catch (IOException e) {
25            e.printStackTrace();
26        }
27    }
28 }
```

The console output at the bottom shows the server starting and waiting for a client connection:

```
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 Socket 编程-启动客户端，服务器端显示内容

```
8 public static void main(String[] args) {
9
10     try {
11         ServerSocket serverSocket = new ServerSocket(8888);
12         System.out.println("10165101180 ***服务器即将启动，等待客户端的连接***");
13         Socket socket = serverSocket.accept();
14         InputStream in = socket.getInputStream();
15         InputStreamReader inreader = new InputStreamReader(in);
16         BufferedReader br = new BufferedReader(inreader);
17         String info = null;
18         while((info = br.readLine())!=null){
19             System.out.println("10165101180 我是服务器，客户端说: "+info);
20         }
21         socket.shutdownInput();
22
23         java.io.OutputStream outputStream = socket.getOutputStream();
24         PrintWriter printWriter = new PrintWriter(outputStream);
25         printWriter.write("10165101180 欢迎您! ");
26         printWriter.flush();
27         socket.shutdownOutput();
28     }
29 }
30
```

Problems @ Javadoc Declaration Console Debug

<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

```
1 import java.io.*;
2
3
4 public class Client {
5     public static void main(String[] args) {
6         try {
7             Socket socket = new Socket("127.0.0.1",8888);
8             OutputStream outputStream = socket.getOutputStream();
9             PrintWriter pw = new PrintWriter(outputStream);
10            pw.write("用户名: 10165101180;密码: 123");
11            pw.flush();
12            socket.shutdownOutput();
13
14            InputStream inputStream = socket.getInputStream();
15            BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
16            String info = null;
17            while((info = br.readLine())!=null){
18                System.out.println("10165101180 我是客户端，服务器说: "+info);
19            }
20            socket.shutdownInput();
21
22            br.close();
23            inputStream.close();
24        }
25    }
26 }
27
```

Problems @ Javadoc Declaration Console Debug

<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

```
1 import java.io.IOException;
10
11 public class RPCServer {
12     private ExecutorService threadPool;
13     private static final int DEFAULT_THREAD_NUM = 10;
14     private ServerSocket server;
15
16     public RPCServer(){
17         threadPool = Executors.newFixedThreadPool(DEFAULT_THREAD_NUM);
18     }
19
20     public void register(Object service, int port){
21         try {
22             System.out.println("10165101180 服务器已启动.");
23             server = new ServerSocket(port);
24             Socket socket = null;
25             while((socket = server.accept()) != null){
26                 System.out.println("10165101180 已连接至客户端.");
27                 threadPool.execute(new Processor(socket, service));
28             }
29         } catch (IOException e) {
30             e.printStackTrace();
31         }
32     }
33 }
```

Problems @ Javadoc Declaration Console Debug

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

```
1 import java.io.IOException;
10
11 public class RPCServer {
12     private ExecutorService threadPool;
13     private static final int DEFAULT_THREAD_NUM = 10;
14     private ServerSocket server;
15
16     public RPCServer(){
17         threadPool = Executors.newFixedThreadPool(DEFAULT_THREAD_NUM);
18     }
19
20     public void register(Object service, int port){
21         try {
22             System.out.println("10165101180 服务器已启动.");
23             server = new ServerSocket(port);
24             Socket socket = null;
25             while((socket = server.accept()) != null){
26                 System.out.println("10165101180 已连接至客户端.");
27                 threadPool.execute(new Processor(socket, service));
28             }
29         } catch (IOException e) {
30             e.printStackTrace();
31         }
32     }
33 }
```

Problems @ Javadoc Declaration Console Debug

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

```
1 import java.io.ObjectInputStream;
7
8 public class RPCClient {
9     public static void main(String args[]){
10         HelloService helloService = getClient(HelloService.class, "127.0.0.1", 50001);
11         System.out.println(helloService.hello("10165101180"));
12     }
13
14     @SuppressWarnings("unchecked")
15     public static <T> T getClient(Class<T> clazz, String ip, int port){
16         return (T) Proxy.newProxyInstance(RPCClient.class.getClassLoader(), new Class<?>[]{c
17
18         private Socket socket;
19
20         @Override
21         public Object invoke(Object arg0, Method arg1, Object[] arg2) throws Throwable {
22             socket = new Socket(ip, port);
23             ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());
24             out.writeUTF(arg1.getName());
25             out.writeObject(arg1.getParameterTypes());
26             out.writeObject(arg2);
27             ObjectInputStream in = new ObjectInputStream(socket.getInputStream());
28             return in.readObject();
29         }
30     };
31 }
32 }
33 }
```

Problems @ Javadoc Declaration Console Debug

<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，客户端和服务端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。