**长沙学院计算机工程与应用数学学院**

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 课程名称 | 面向对象程序设计 | 班级 | 20软件 | 实验日期 | 4月25日 |
| 姓名 | 李佳骏 | 学号 | B20190103224 | 指导教师 | 黄志坚 |
| 成绩 |  | | | | |

一. 实验名称

JAVA网络编程

二. 实验目的及要求

1）熟练通过TCP编码完成服务器端和客户端的代码编写，实现1对1通信

三. 实验环境

IDEA jdk1.8

四. 实验内容

（1）实现客户端文件上传至服务器端，将包下的data.txt文件传输至服务器（命名为config.imi）

五. 实验过程及实验结果

1. 编写创建Client客户端java代码

*package* classroom;  
  
*import* java*.*io*.*\*;  
*import* java*.*net*.*InetAddress;  
*import* java*.*net*.*Socket;  
  
*public class* Client {  
 *public static void* main(String[] args) *throws* IOException {  
 *//指定接收地址和端口号  
// System.out.println(InetAddress.getLocalHost());* Socket socket = *new* Socket(InetAddress*.getByName*("count"), 10086);  
  
 BufferedReader br = *new* BufferedReader(*new* InputStreamReader(*new* FileInputStream("/home/ljjtpcn/Desktop/work/CS学科/java实验工程代码(IDEA)/网络编程/src/classroom/data.txt")));  
 BufferedWriter bw = *new* BufferedWriter(*new* OutputStreamWriter(socket*.*getOutputStream()));  
  
 String line;  
 *while* ((line = br*.*readLine()) != *null*) {  
 bw*.*write(line);  
 bw*.*newLine();  
 bw*.*flush();  
 }  
 socket*.*shutdownOutput();  
  
 BufferedReader response = *new* BufferedReader(*new* InputStreamReader(socket*.*getInputStream()));  
 System*.*out*.*println("服务器回应:" + response*.*readLine());  
 response*.*close();  
 br*.*close();  
 bw*.*close();  
 }  
}

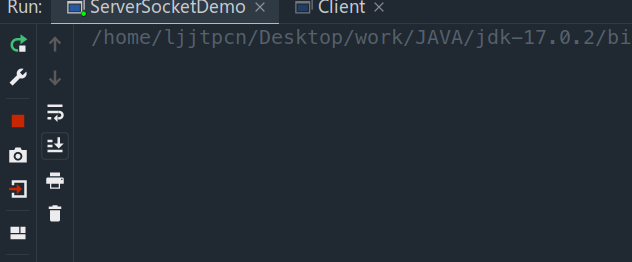
1. 编写服务器端代码

*package* classroom;  
  
*import* java*.*io*.*\*;  
*import* java*.*net*.*ServerSocket;  
*import* java*.*net*.*Socket;  
  
*public class* ServerSocketDemo {  
 *public static void* main(String[] args) *throws* IOException {  
 ServerSocket ss = *new* ServerSocket(10086);  
 Socket accept = ss*.*accept();  
  
 BufferedReader br = *new* BufferedReader(*new* InputStreamReader(accept*.*getInputStream()));  
 BufferedWriter bw = *new* BufferedWriter(*new* FileWriter("/home/ljjtpcn/Desktop/work/CS学科/java实验工程代码(IDEA)/网络编程/src/classroom/config.imi"));  
  
 String line;  
 *while* ((line = br*.*readLine()) != *null*) {  
 bw*.*write(line);  
 bw*.*newLine();  
 bw*.*flush();  
 }  
  
 BufferedWriter bufferedWriter = *new* BufferedWriter(*new* OutputStreamWriter(accept*.*getOutputStream()));  
 bufferedWriter*.*write("上传成功！");  
 bufferedWriter*.*newLine();  
 bufferedWriter*.*flush();  
  
 bufferedWriter*.*close();  
 bw*.*close();  
 br*.*close();  
 }  
}

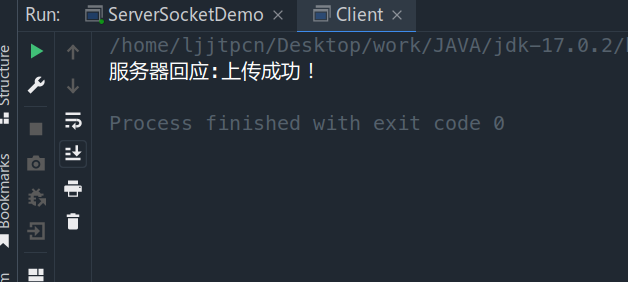
1. 测试
2. 首先允许服务器端程序开启服务器
3. 运行客户端程序，观察文件上传是否成功
4. 成功判断标准为是否收到服务器正反馈且包下是否有新文件创建成功

4. 运行截图

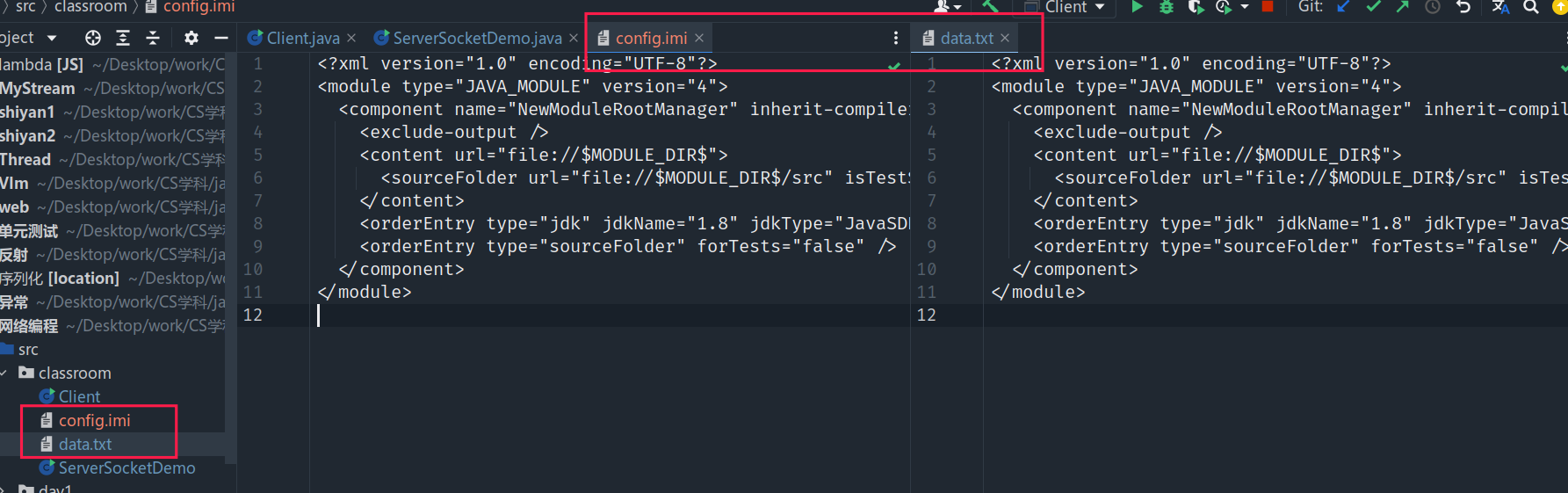
服务端



客户端



上传成功



1. 实验小结
2. TCP协议和UDP协议是传输层的两种协议。Socket是传输层供给应用层的编程接口，所以Socket编程就分为TCP编程和UDP编程两类。
3. TCP一种面向连接（连接导向）的、可靠的、基于字节流的运输层（Transport layer）通信协议
   1. 面向连接
   2. 点到点通信
   3. 高可靠性：三次握手
   4. 占用系统资源多，效率低

生活案例： 打电话

1. UDP 一种无连接的传输层协议，提供面向事务的简单不可靠信息传送服务
   1. 非面向连接， 传输不可靠， 可能丢失
   2. 发送不管对方是否准备好， 接收方收到也不确认
   3. 数据报的大小限制在 64k 内
   4. 非常简单的协议，开销小

生活案例：发送短信 发电报

1. 在Java环境下，Socket编程主要是指基于TCP/IP协议的网络编程。 Server端Listen(监听)某个端口是否有连接请求，Client端向Server端发出Connect(连接)请求，Server端向Client端发回Accept（接受）消息。一个连接就建立起来了。Server端和Client端都可以通过Send，Write等方法与对方通信。
2. 网络编程的核心是IP、端口、协议三大元素
3. 网络编程的本质是进程间通信
4. 网络编程的2个主要问题：1是定位主机，2是数据传输

七. 教师评阅意见