**计算机网络编程**

**实验报告**

**班级：07111707**

**组长：1120171189 崔程远**

**成员：1120172149 吴沁璇**

**1120172153 张澈**

**1120172163 王晓媛**

**1120172736 张鉴昊**

**1120172765 曾煜瑾**

**1120173326 曾紫飞**

**北京理工大学**

**计算机学院**

**2020年5月**

**第六章 实验2 TCP协议服务器和客户**

**1. 实验目的**

学习TCP通信协议，掌握TCP socket通信方法。

1. **实验内容**

客户发送命令行文本给服务器，服务器转换大写后返回给客户并显示。

配置文件关键要点：

无，对方的IP地址、端口以及发送串以命令行参数的形式提供程序运行

1. **实验原理**

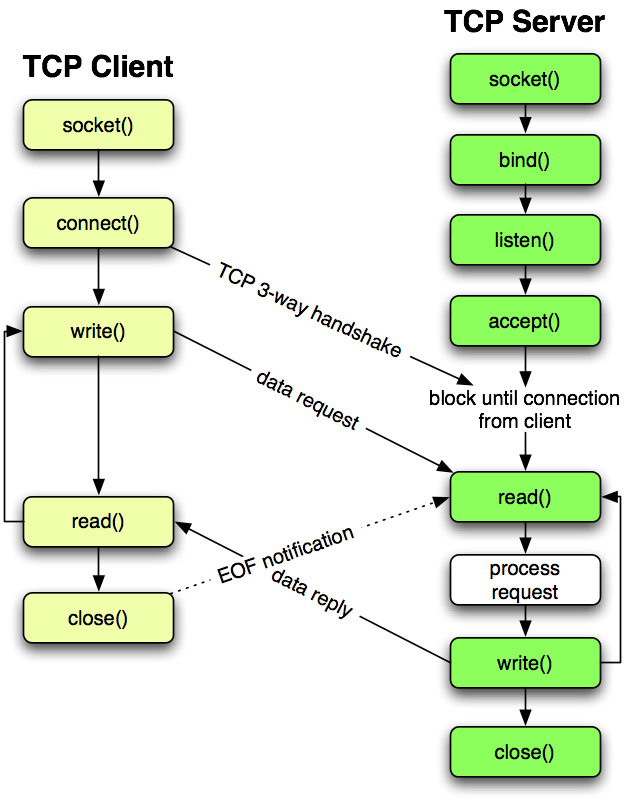
socket起源于Unix，而Unix/Linux基本哲学之一就是“一切皆文件”，都可以用“打开open –> 读写write/read –> 关闭close”模式来操作。我的理解就是Socket就是该模式的一个实现，socket即是一种特殊的文件，一些socket函数就是对其进行的操作（读/写IO、打开、关闭）。

TCP编程的服务器端一般步骤是：

1. 创建一个socket，用函数socket()；
2. 绑定IP地址、端口等信息到socket上，用函数bind();
3. 开启监听，用函数listen()；
4. 接收客户端上来的连接，用函数accept()；
5. 收发数据，用函数send()和recv()，或者read()和write();
6. 关闭网络连接；
7. 关闭监听；

TCP编程的客户端一般步骤是：

1. 创建一个socket，用函数socket()；
2. 绑定IP地址、端口等信息到socket上，用函数bind();\* 可选
3. 设置要连接的对方的IP地址和端口等属性；
4. 连接服务器，用函数connect()；
5. 收发数据，用函数send()和recv()，或者read()和write();
6. 关闭网络连接；

图源网络

1. **实验环境**

|  |  |  |
| --- | --- | --- |
| 语言 | 集成开发环境 | 编译器 |
| C++ | Visual Studio 2017 | gcc version 4.8.1 |
| Java | Eclipse 2019 | java version "1.8.0\_65" |
| Python | Pycharm 2017 | Python 3.7.0 |

1. **实验步骤**

各个语言的实现都分为服务端和客户端。

**·C语言版本**

服务端首先判断接收参数数量是否符合要求，之后创建Socket并绑定到本地端口；通过listen()阻塞等待连接；通过accept()阻塞等待；通过recv()接收信息；处理后通过send()发送；关闭Socket。

客户端首先判断接收参数数量是否符合要求，之后创建Socket；通过connect()连接服务端；send()发送字符串;recv()接收处理后的字符串；关闭Socket。

**·Java语言版本**

服务端首先判断接收参数数量是否符合要求，之后创建ServerSocket对象完成绑定；通过ServerSocket对象的accept()方法创建Socket对象；通过Socket对象获得输入输出流InputStream OutputStream对象；通过输入输出流对象获得待处理信息并将处理后信息发送；关闭输入输出流和Socket。

客户端首先判断接收参数数量是否符合要求，之后创建Socket对象；通过Socket对象获得输入输出流InputStream OutputStream对象；通过输入输出流对象发送待处理信息并获得处理后信息；关闭输入输出流和Socket。

**·Python语言版本**

服务端首先判断接收参数数量是否符合要求，之后创建Socket并绑定到本地端口；通过listen()阻塞等待连接；通过accept()阻塞等待；通过recv()接收信息；处理后通过send()发送；关闭Socket。

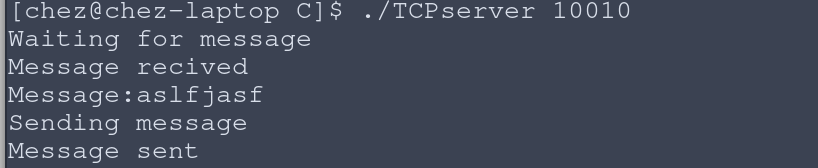
客户端首先判断接收参数数量是否符合要求，之后创建Socket；通过connect()连接服务端；send()发送字符串;recv()接收处理后的字符串；关闭Socket。

1. **运行结果**

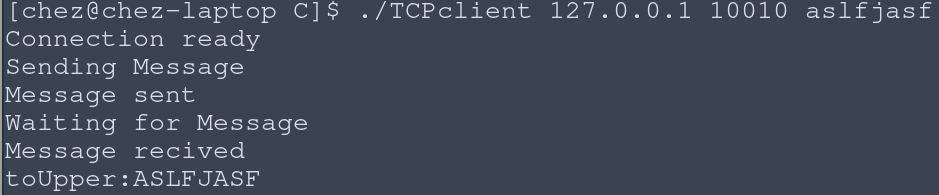
Send函数中将01100000000000111110000000000110作为待发送的数据信息。

**·C**

服务端：输入开放的端口号

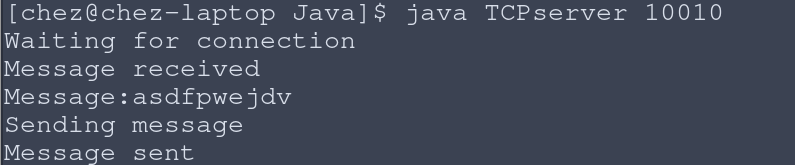


客户端：依次输入目标IP、目标端口、待处理字符串

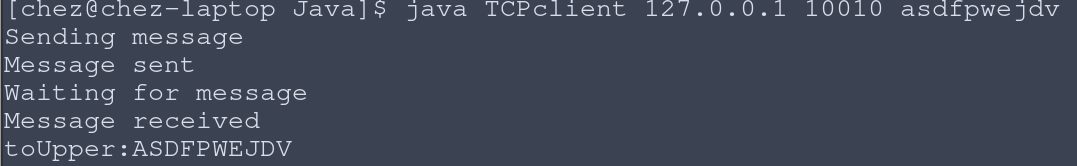


**·Java**

服务端：输入开放的端口号

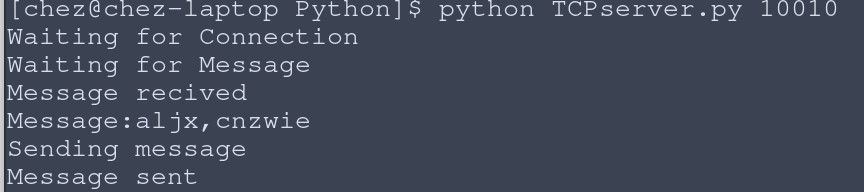


客户端：依次输入目标IP、目标端口、待处理字符串

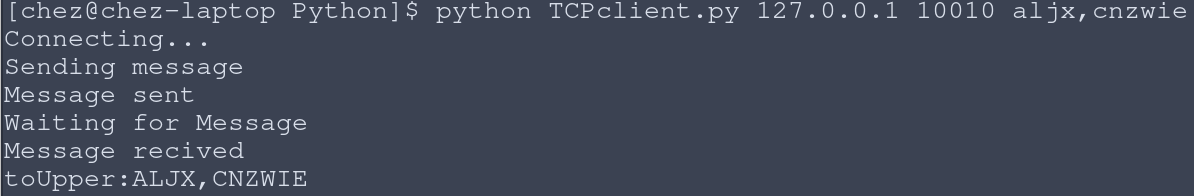


**·Python**

服务端：输入开放的端口号



客户端：依次输入目标IP、目标端口、待处理字符串



1. **实验总结**

实验内容相对简单，通过上一个实验后基本熟悉了Socket编程方法， TCP Socket编程和UDP相比有所不同，但了解了各个语言实现TCP通信的基本函数和类后还是能很快完成功能。由于各种语言对UDP socket的实现机制不同，不同系统对socket的支持也不同（POSIX标准和Winsocket），因此需要查阅相关资料。

总的来说，这个实验使我加深了对TCP通信过程的理解，提高了编程能力。Socket极大地方便了网络编程。