

网络应用程序设计实践

1. 实践目的

- ✚ 写出本次实践的目的，要求简洁清楚。
- ✚ 目的是为了让学生加深理解计算机网络的工作方式，计算机网络的体系结构和 TCP/IP 协议栈，学会使用常用的网络命令并能够利用这些命令进行网络故障的检测；该实践课程还让学生在熟悉 Eclipse 开发环境的基础上，使学生掌握 Java 网络编程、多线程编程和 RMI 编程，能够熟练掌握面向对象的方法，使用 Java 的编码规则，能够利用所学的计算机网络知识和 Java 编程语言知识根据需求进行程序设计，独立完成程序设计，培养学生理论联系实际的能力。

2. 预习内容

- ✚ 写出本次实践过程中你所用到的知识；
- ✚ (1) FTP 通信流程和通信命令 FTP 使用 2 个端口，一个数据端口和一个命令端口（也叫做控制端口）。这两个端口一般是 21（命令端口）和 20（数据端口）。
 - 1> 命令端口 一般来说，客户端有一个 Socket 用来连接 FTP 服务器的相关端口，它负责 FTP 命令的发送和接收返回的响应信息。一些操作如“登录”、“改变目录”、“删除文件”，依靠这个连接发送命令就可完成。
 - 2> 数据端口 对于有数据传输的操作，主要是显示目录列表，上传、下载文件，我们需要依靠另一个 Socket 来完成。
- ✚ (2) Socket 编程 服务器端创建一个 serversocket，然后调用 accept 等待客户端连接，客户端建立一个 socket 与服务器端连接。
- ✚ (3) 多线程编程 自己编写一个类继承自 Thread 类，并添加 socket 属性，写入新的构造函数，以及对 run 方法进行重写。
- ✚ (4) HTTP 通信原理 http 的请求格式，以及应答格式
- ✚ (5) JAVA GUI 编程 使用 Eclipse 中的 WindowBuilder 插件或 netbeans 的拖拽功能实现 GUI 界面。
- ✚ (6) 综合运用 Socket, 多线程, GUI 编程实现多人即时通信功能

3. 实践内容和实践过程

1. 编写简单的 HTTP 1.1 客户端程序；编写简单的 HTTP 1.1 服务器程序；
 - (1) 实践内容

本实验要求完成以下功能（使用 GET 方式）：

服务器功能 提供 HTML, JPG 等 MIME 类型的资源

客户端功能	访问服务器，获取 HTML 和 JPG 资源，保存到本地磁盘
	访问服务器，获取 WMV 等其他资源类型，保存到本地磁盘

(2) 实现过程

注意：

- ① 具体体现 **Socket** 的编程流程；
- ② 具体体现 **HTTP** 协议的通信流程；
- ③ 具体体现使用多线程实现 **HTTP1.1** 服务器同时处理多个 **HTTP1.1** 客户端的实现方法；
- ④ 最好利用 **UML** 的类图或结构化的程序流程图来描述程序的实现过程；

(3) 关键技术

关键写出在完成各个功能时所使用的具体函数，但不要将全部程序代码粘贴；

(4) 遇到的问题及解决方案

第一个问题，之前在创建文件的时候程序一直报错，说文件路径不对或者不存在，之后发现在写路径的时候 '/' 这些元字符是需要转义的。

第二个问题，是服务器运行后，无法在网页上查看图片等信息，同时服务端抛出异常，但是 **html** 网页可以正常显示。这个问题最后通过询问同学和查资料，得知我所使用的发送和接收消息的流不对，不能够使用 **datainputstream** 和 **dataoutputstream**，并且 **wirteUTF** 方法不能用于 **http** 服务器中因为这个方法一开始会写入两个字节，即为将要写入的字节数（不是字符串的长度）。后来改成了 **printstream** 和 **bufferedReader** 就成功了。

第三个问题，是没有意识到浏览器解析 **html** 时，对于 **html** 内嵌的图片和视频等资源会在解析之后再次送一个 **get** 请求。所以通过浏览器访问服务端网页时，服务端不用进行标签解析。

第四个问题，在服务器发送响应体的时候，我发现出现一个奇怪的现象，就是程序运行的时候，总是无法下载完成，打开对应的客户端文件夹发现文件没有下载完成（即为 0 字节），但是在关闭客户端运行之后，那个文件夹里的文件就下载好了。最后发现，是服务器在发送响应体的时候，最后忘记关闭了文件发送流，所以导致客户端一直在读取服务器的信息，导致失败。后来添加了 **close**，判断 -1 才能成功，接受所有信息并关闭文件。


2. 编写简单的 FTP 客户端程序；编写简单的 FTP 服务器程序

(1) 实践内容

获取文件列表	给定用户工作目录, 获得文件列表, 包括文件名称, 文件大小, 文件创时间
文件上传	上传文件到用户指定工作目录中


	上传文件夹到用户指定工作目录中
	支持断点续传
文件下载	给定用户工作目录, 下载文件
	给定用户工作目录, 下载文件夹
	支持断点续传

(2) 实现过程

 注意:

- ① 具体体现 FTP 协议的通信流程;
- ② 具体体现使用多线程实现 FTP 服务器同时处理多个 FTP 客户端的实现方法;
- ③ 最好利用 UML 的类图或结构化的程序流程图来描述程序的实现过程;

(3) 关键技术

 关键写出在完成各个功能时所使用的具体函数, 但不要将全部程序代码粘贴;

(4) 遇到的问题及解决方案

第一个问题是很迷茫, 应为是第一个做的实验, 不知道如何进行文件上传和下载。后来在网上查询, 明白了可以分成两部分, 第一部分把文件从硬盘里读取到内存中, 再把数据从内存中发送到另一端, 由另一端读取到内存, 再从内存写入到文件。其中使用了一个字节数组当做一个缓冲, 就像一辆运输车一样, 一趟一趟的运送货物, 直到最后 `read() == -1` 就读完了。

第二个问题是下载文件夹。逻辑较为复杂, 首先要使用 `isDirectory` 方法判断是否是文件夹, 然后在本地判断是否存在同名文件夹, 通过 `file.list` 方法获得文件列表, 进而得到文件个数, 然后对每个文件进行下载即可, 此处隐含了 `list`, `cwd`, `retr` 等指令, 较为复杂, 并且 `dos` 使用使用后要及时 `flush` 和关闭。

第三个问题是阻塞方法的调用。多线程间的协作和 `socket` 通信容易混淆, 了解到相关原则是尽量少开线程, 并时刻注意线程同步异步问题, 如上传文件时, 服务端使用线程同步

(`synchronized`) 的方法, 即一个线程访问该方法时, 他就获得了该 `object` 的对象锁, 另一个线程对该 `object` 的访问将被暂时阻塞, 保证只存在一个同名文件且后上传的不会覆盖之前的。由于实验末尾才稍微了解了 `java` 并发的锁机制, 所以所有实验都可以用线程锁等 `java` 并发编程知识进行优化, 这也是下一步的改进方向。


3. 编写即时通信系统的客户端程序; 编写即时通信系统的服务器程序

(1) 实践内容

用户登录: 用户登录。

即时通讯: 查看用户列表, 客户端之间互相发送消息。

(2) 实现过程

 注意:

- ① 具体体现客户端和服务端通信的通信协议；
- ② 具体体现如何使用多线程实现一个服务器同时处理多个客户端的过程（多人同时聊天）；
- ③ 具体体现如何使用多线程实现单个客户端同时可以收消息和发消息；
- ④ 具体体现聊天界面的页面元素设计和控件事件的处理；
对于非网络技术核心的表情，聊天字体，上线提醒，添加好友等功能，有些因为时间紧迫只有初步实现。
- ⑤ 最好利用 UML 的类图或结构化的程序流程图来描述程序的实现过程；

(3) 关键技术

- ✚ 关键写出在完成各个功能时所使用的具体函数，但不要将全部程序代码粘贴；

4. 遇到的问题及解决方案

第一个问题是对该设计模式的理解。这个模型类似于生产-消费者模型，将该模型的订阅模式和 p2p 模式结合，就实现了即能群聊，又能单独聊天的功能。一开始看通信流程图，认为是每个用户一个消息信道，直到想明白生产-消费者模型，才开始正确的实验过程。网上相关资料实现了 jms 接口进行开发，这也是可以进一步改进的方向

第二个问题是初始化用户列表的时机，后来发现了问题，添加了构造函数，初始化了用户列表。但是主线程里并没有初始化任何一个用户包对象，仅仅是调用了一个静态方法，所以我在服务器建立伊始就初始化了一个用户包对象，虽然这个对象没有被使用，但是成功初始化了用户列表。同样的，消息列表类（MessageDAO）我也使用了此种方法，只不过不在建立服务器时，是在用户登录成功过后。

4. 编写 RMI 程序(可选)；

(1) 实践内容

实现分布式议程服务，使用文件等进行会议存储。

(2) 实现过程

- ✚ 注意：

- ① 具体体现 RMI 的交互流程；
- ② 在此实践中类的设计、某些功能模块的实现流程；
- ③ 最好利用 UML 的类图或结构化的程序流程图来描述程序的实现过程；

(3) 关键技术

- ✚ 关键写出在完成各个功能时所使用的具体函数，但不要将全部程序代码粘贴；

(4) 遇到的问题及解决方案

最大的问题是对于远程调用的理解。本地没有实现任何接口中的方法，通过调用远程服务端（不同的 jvm）的程序，传递参数和对象，达到和本地运行一样的效果。对于 stub，即注册存根的方法通过网上的教程和实例了解了一些，相当于本地这侧在远程

机器的存根，服务端改变本地也要改变。之后就是正常的读写服务端文件操作来实现议程服务了。

4. 实践总结

具体体现本次课程实践的收获，以及对实践课的意见和建议。

技术上的提升：

1. 加深了 HTTP 协议，FTP 协议的理解；
2. 熟练使用 JAVA 的多线程，网络编程，GUI 编程实现通信功能；
3. 提升了 JAVA 编程的动手和调试能力
4. 熟悉了 UML 图的画法；

其他方面的提升：

1. 通过课程实践提升了网络/软件编程的兴趣；
2. 通过答辩，跟老师和同学交流问题，提升了沟通表达的能力；

建议：就是感觉时间有点少，任务有点复杂。做了这次实践我感觉只要坚持去写，坚持查资料，再难的问题也能一一解决。像这次实验，刚开始，我觉得非常难而且无从下手，一点都不会写。但是只要静下心，慢慢写，慢慢上网查，问题总是能一一解决的。可能是我没有学习设计模式，再加上时间没有好好把握，所以才会出现很多问题耽误了时间。但是经过这次实验，我的学习能力得到了提高，面对困难的问题也不会像刚开始这样害怕了。我也希望以后的实验能完成得越来越好

5. 参考资料

列出实践过程中使用到的参考文献列表。

计算机网络（第 6 版）谢希仁

Java 从入门到精通（第 3 版）明日科技 编著