**基于Python Socket实现的简单的FTP Server**

**服务器实现**

服务端主要由FTPServer, Server, DataSocket, IpListManagement, WebSocket几个类组成。

**Server(WebSocketServer)**

**程序的主要类，负责与单个客户端进行交互，并继承线程，用以实现多线程处理。**

**主结构由一个大的While循环组成，每当收到来自客户端的请求报文时，解析报文，查看报文包含的命令，并依次比对进行处理。**

**由于防火墙问题，不适用PORT传输模式。**

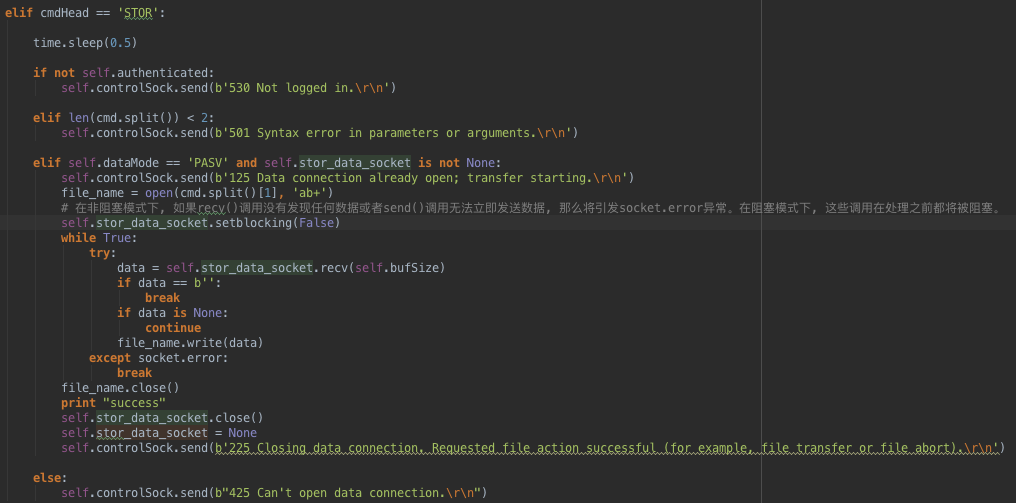
**controlSock对应control socket，用以进行命令的交互。通过PASV命令设置被动数据传输模式，并获取客户端用以连接服务端的地址和接口。**

**Control Socket的相应报文均符合RFC959要求。**

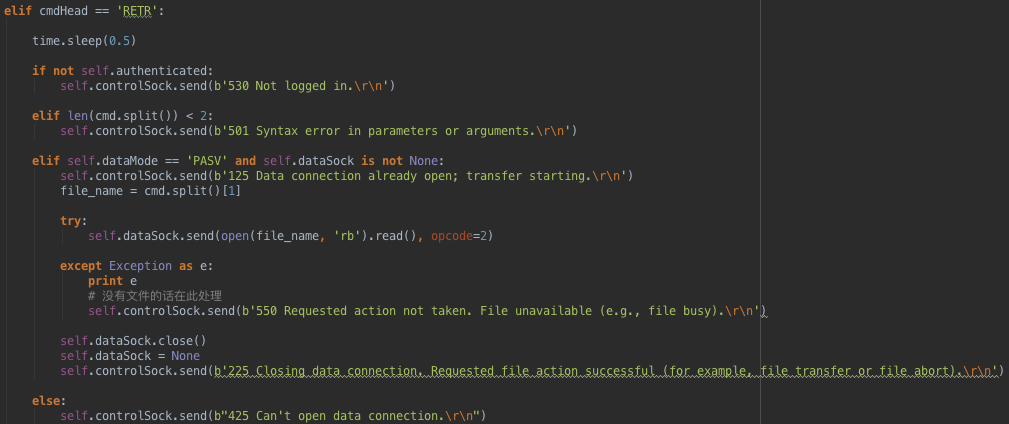
**dataSock对应data socket用以进行数据传输，每次数据传送完成则关闭该socket。**

**dataListenSock 用于监听用户发起的data socket连接。**

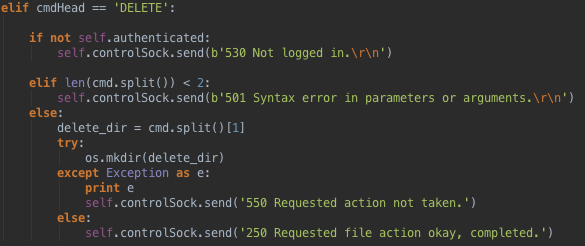
**Server 实现了主要的基础功能包括上传STOR：**

****

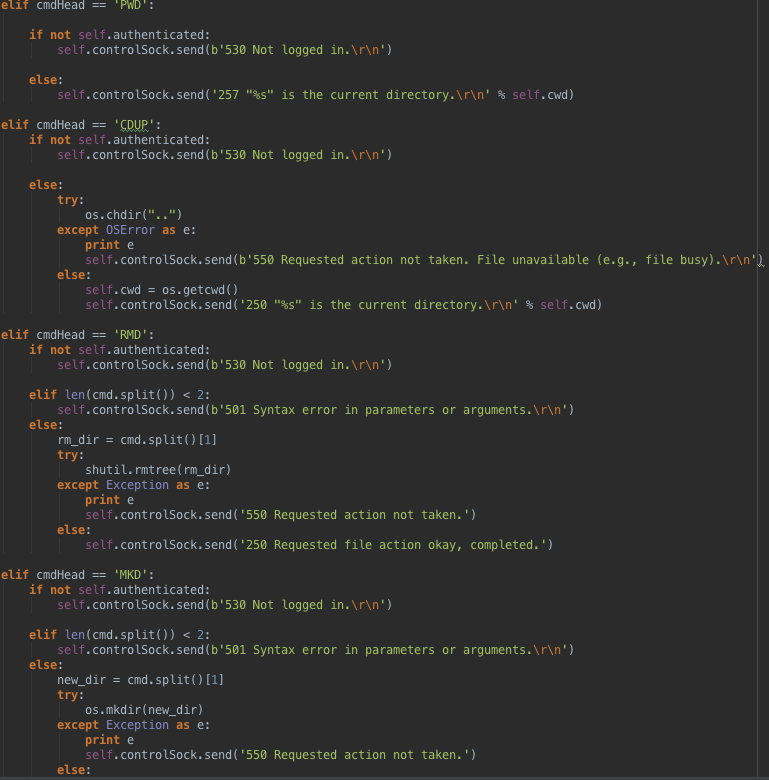
**下载RETR：**

****

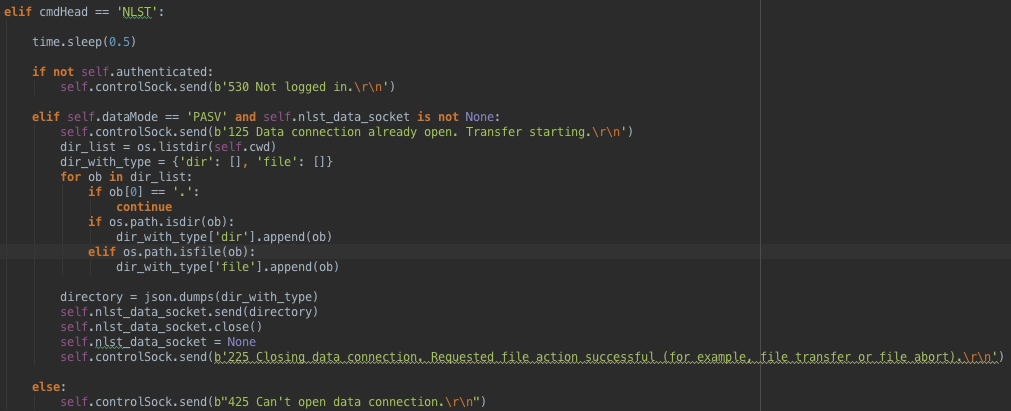
**删除DELETE:**

****

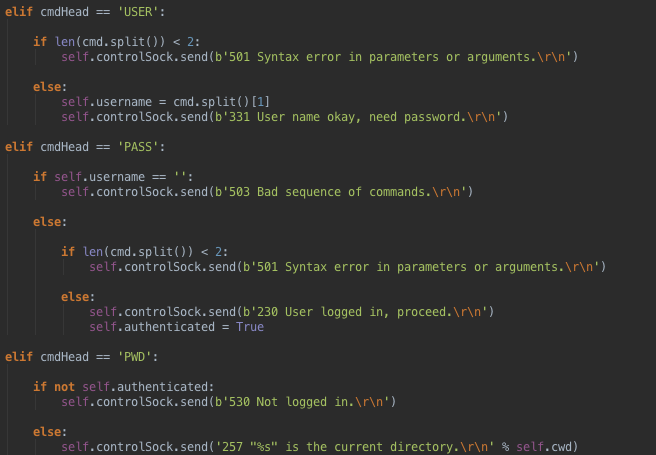
**目录相关操作的PWD, CDUP, RMD, MKD：**

****

**获取文件列表NLST:**

****

**以及用以用户认证的USER和PWD：**

****

**WebSocket是对socket进行的再封装。客户端使用web界面，通过JavaScript的WebSocket与服务器进行交互。**

**FTPServer:**

**对于连接上的客户端通过请求端口判断是来自WebSocket还是来自primitive socket，然后分配给不同的监听socket。**

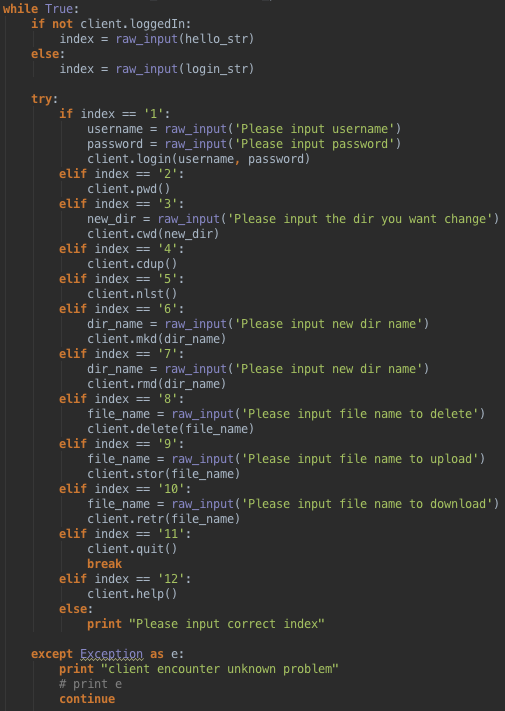
****

**客户端实现**

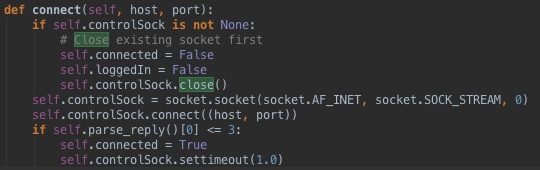
**Primitive Client:**

**通过原生的socket实现的FTPClient。**

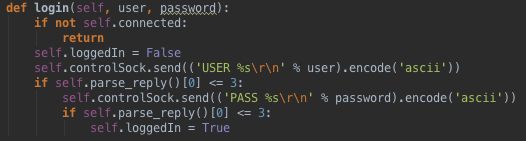
**主程序入口采用简单的列表式功能选择：**

****

**程序运行后首先会调用connect连接服务器：**

****

**选择登陆功能登陆后才可以再选择其他功能：**

****

**详细代码参阅client.py文件**

**参考资料**

**RFC959 FTP protocol 文档**

**WebSocket 协议文档**