计算机通信网络大作业项目文档

--FTP文件传输

516030910284 马兴华 516030910287 裘炜程

1. FTP传输的原理介绍及程序设计思想
2. 前期准备工作
3. 开发环境选择

在商议之后，我们选择在Windows系统上使用Python3进行开发，

因为Python语言简单强大，还有很多强大的扩展库可以简化编程操作。我们安装了Visual Studio，使用Python 3.6.4作为开发环境，并在github上创建了仓库进行代码同步和版本管理。

1. Socket套接字编程原理

在传输层中，每一条 TCP 连接有两个端点，TCP 连接的端点就

叫做套接字（socket）。端口号拼接IP 地址即构成了套接字（套接字 socket = (IP地址: 端口号)）。每一条 TCP 连接唯一地被通信两端的两个端点（即两个套接字）所确定，即TCP 连接可表示为：

{socket1, socket2}= {(IP1: port1), (IP2: port2)}

在服务器端，套接字的运行过程是：首先，服务器端使用socket函数和bind函数，绑定一个IP地址和对应的端口号，然后，调用listen函数监听这个端口，当收到客户端发来的connect请求时，调用accept函数来接受客户端请求，连接成功建立，开始数据传递阶段，实际上就是TCP中建立连接的三次握手的函数化实现。

在客户端，套接字的运行过程是：客户端调用socket函数，建立一个socket实体，然后调用connect函数向服务器发送连接请求，服务器会接受这次连接请求，然后有系统自动分配一个端口号和自身的IP地址组合，双方建立连接，进入数据报文传输阶段。

但是在本项目中，没有使用Socket库，而是选择使用了更高级别的封装了ftp功能的扩展库，这些扩展库利用Socket编程的原理在更高层次上实现了ftp的功能。

3.使用的Python标准库介绍

1. PyQt5

由于这个项目需要实现用户友好的界面，所以需要图形化编程，原

来都是使用Tkinter的，但是界面和按钮太多编写太麻烦，于是选择了PyQt5作为界面化生成工具。与其说是一个扩展库，不如说是一个Python实现的小工具，让你用界面化图形化的方式编辑页面，可以选择页面大小，拖拽按钮、文本显示、listbox等控件进行编辑，还可以选择按钮的触发函数，保存会生成.ui类型的文件，然后使用pyuic5.py可以生成对应的py文件，即可添加到项目代码里。

PyQt5运行截图

1. ftplib

ftplib是Python中内置的一个实现了ftp客户端功能的扩展库，其

中用到的封装的主要函数有：

1）login(user='',passwd='', acct='') 登录到FTP 服务器

2）pwd() 当前工作目录

3）cwd(path) 把当前工作目录设置为path

4）nlst([path[,...])返回一个文件名的列表

5）retrlines(cmd [, cb]) 给定FTP 命令（如“RETR filename”），用于下载文本文件。可选的回调函数cb 用于处理文件的每一行

6）retrbinary(cmd, cb[,bs=8192[, ra]]) 与retrlines()类似，只是这个指令处理二进制文件。回调函数cb 用于处理每一块（块大小默认为8K）下载的数据。

7）storlines(cmd, f) 给定FTP 命令（如“STOR filename”），以上传文本文件。要给定一个文件对象f

8）rename(old, new) 把远程文件old 改名为new

9）delete(path) 删除位于path 的远程文件

10）mkd(directory) 创建远程目录

这些封装函数极大地简化了我们对客户端的编程工作。

1. pyftpdlib

pyftpdlip库是一个专门用于编写ftp服务器端的扩展库，其中提供

了很多强大的类和函数来解决问题，主要使用了authorities类中的DummyAuthorizer函数用于创建新的服务器实体，handlers中的

ThrottledDTPHandler用于对服务器的上传和下载限速，FTPServer类和FTPHandler类处理连接、下载、上传等ftp业务。

使用pyftpdlib库，简化了对服务器的编码。

1. pyinstaller

由于要打包生成可执行文件（.exe）类型的文件，我们选择

pyinstaller扩展库实现这个功能。将所有需要的资源文件打包放入同一个文件夹后，使用生成命令，pyinstaller的生成命令是：

pyinstaller –command file.py

其中可用的command指令有：

-F 表示生成单个可执行文件

-w 表示去掉控制台窗口

-p 表示自定义需要加载的类路径

-i 表示可执行文件的图标

我们成功地生成了可执行文件并成功在服务器上运行。

1. 开发过程及遇到的问题

第一阶段我们主要做server和client的基础功能，然后设计了友好的用户界面，并与功能相连接。

1. 程序的算法与结构

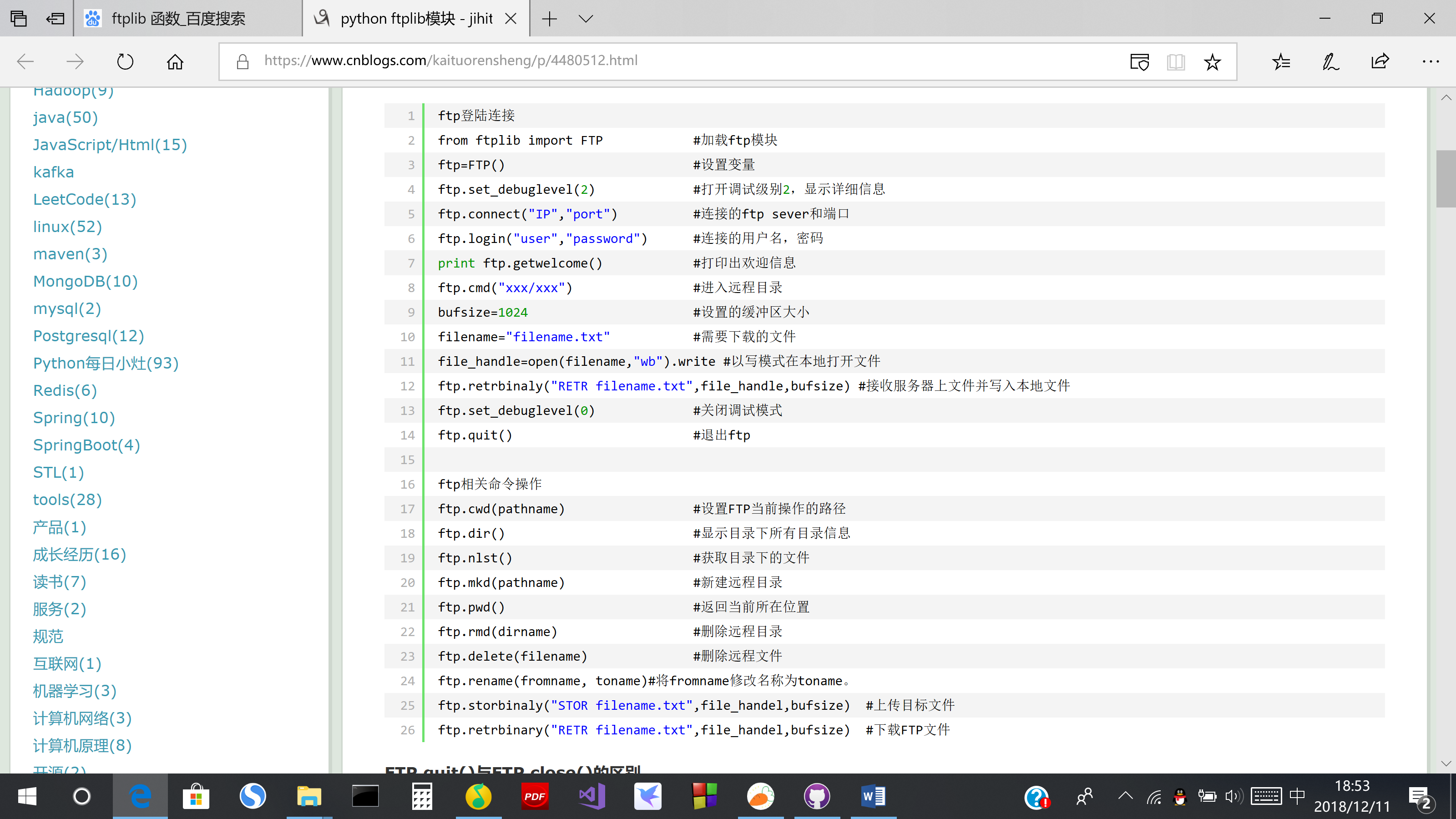
服务器：

所有功能以pyftpdlib中的函数为基础，我们将建立服务器的过程封装在一个函数init\_ftp\_server()中，所在的py文件为ftp\_server.py，包括DummyAuthorizer函数用于创建新的服务器实体，ThrottledDTPHandler用于对服务器的上传和下载限速，FTPServer类和FTPHandler类处理连接、下载、上传等ftp业务。其中会用到很多参数，这里统一存储在config\_ftp.py中。登陆时用到了用户信息，储存在userinfo.ini中，每次打开服务器会将这里面的信息读取出来，为后续的用户信息处理做准备，最后将上述的函数合并成ftp()函数。

窗口界面的设计上，我们用了pyqt5来设计，主窗口为server.py，点击“添加新用户”，弹出create框，对应create.py，用于添加新的用户信息；点击“修改账户”，弹出change框，对应change.py，用户修改已有账户信息；点击“开启服务器”，运行服务器，直接调用ftp()函数并关闭主窗口；点击“退出程序”就关闭主窗口；下半部分是信息的显示，可以自动显示服务器IP地址和开放的端口号（由于我们选择的是被动连接），调用get\_ip.py中的get\_host\_ip()函数。在主程序test.py中，我们定义了窗口按钮与功能连接的函数，在进入create框后，按提示填入信息，点击“创建新用户”，调用我们定义的create()函数，进行检查输入信息、创建用户、记录信息；在进入change框后，按提示修改信息，点击“修改”，调用我们定义的change()函数，进行检查修改信息、useinfo.ini修改信息。

客户端：

整个主要功能以ftplib中的client端的函数为基础，我们在这里完整的封装了一个客户端类ftpclass，在Ftp\_class.py中。核心的函数及功能如下：



在窗口显示方面，我们用pyqt5库做出了可视化窗口，其中第一部分是用户登录功能，用户能够输入服务器主机地址，用户名及密码，然后按登录键登录，该按钮的连接函数是login。第二部分是显示状态，比如登录成功的信息、下载文件成功失败的提示信息等等，这里在程序中是一个textBrowser框，所有功能函数会输出一些提示信息显示在这个框中。第三个部分是显示文件名，在窗口中是一个listView框，每一次功能的操作如果需要显示文件名就会显示在这个框中。第四个部分是一个textBrowser框，用来显示当前文件夹的详细信息。第五个部分是8个功能按键，连接的功能函数分别为返回上一级back、进入下一级enter、 文件重命名rename、创建新文件夹create、上传文件upload、下载文件download、删除文件delete、退出客户端close。除此之外还增加了一些功能：上一次的登陆信息会保留下来，下一次再打开客户端后这些信息会显示在框中，以便快速登陆；双击文件可以进入文件夹；上传下载后会显示上传和下载的文件多大、用了多久。将主窗口的函数与我们定义的Ftp\_class类（以下简称F）的函数连接起来，就可以得到理想的功能：

1. 点击登陆-执行login：显示欢迎信息、将框中信息读入、调用F类中connect连接服务器、文件显示、登陆信息写入info文件（为下一次登陆做准备）。
2. 点击上一级-执行back：调用F类中的changecwd、文件显示。
3. 点击下一级-执行enter：识别鼠标点在哪个文件夹上、调用F类中的changecwd、文件显示。
4. 点击文件重命名-执行rename：识别鼠标点在哪个文件夹上、调用F类中的rename、文件显示。
5. 点击创建新文件夹-执行create：调用F类中的mkdir、在弹窗中输入文件夹名、文件显示。
6. 点击上传文件-执行upload：调用F类中的unpload、在弹窗中选择本地文件、文件显示。
7. 点击上传文件-执行download：识别鼠标点在哪个文件上、调用F类中的download、文件显示。
8. 点击删除文件-执行delete：识别鼠标点在哪个文件上、调用F类中的dellete、文件显示。
9. 点击退出-执行quit：窗口关闭、连接断开。
10. 双击文件名-执行clicked：调用enter。
11. 程序测试截图及说明
12. 体会与建议