

实验目的：

本项目要求设计一个具体的协议（建议是应用层协议），采用标准 SocketAPI 编程来实现协议的功能。在给出的三个题目中，我选择了设计一个简易的小说阅读器。

设计过程：

- 1 首先我根据 PPT 中给出的 echo-client 和 echo-server 以及多线程的设计方法实现了一个基本的多线程客户端和服务端连接功能（见 Socket 文件夹）。
- 2 然后设计服务器的功能，根据客户端传递的请求做出响应（见 Server 文件夹内）。
- 3 最后进行客户端和前端的设计以及与后端的连接（见 Client 文件夹）。

具体描述：

- 1 服务器启动时创建一个套接字，绑定套接字到本地的 IP 端口，并开始监听连接，本服务器是一个多线程服务器，可以同时支持多个客户端的请求，每传入一个连接就启动一个子线程来处理。

线程池的管理：

```
class ThreadPoolManger():
    def __init__(self, thread_num):
        self.work_queue = Queue()
        self.thread_num = thread_num
        self.__init_threading_pool(self.thread_num)

    def __init_threading_pool(self, thread_num):
        for i in range(thread_num):
            thread = ThreadManger(self.work_queue)
            thread.start()

    def add_job(self, func, *args):
        self.work_queue.put((func, args))

class ThreadManger(Thread):
    def __init__(self, work_queue):
        Thread.__init__(self)
        self.work_queue = work_queue
        self.daemon = True
```

2 服务器根据客户端的请求内容处理请求，具体协议设计如下：

请求	行动
0 [filename]	下载名为 filename 的文件（.db 和.jpg 文件）
1 [filename]	打开文件名为 filename 的文件（文本文件）
2 [num]	跳转到第 num 页
3	关闭文本阅读器
其他	返回 invalid request

部分代码如下：

```

while True:
    try:
        rq = conn.recv(buffsize).decode('utf-8')
        print('request:', rq)

        if rq == '':
            break

        if rq[0] == '0':
            filename = rq[2:]
            print("the client is asking for a file", filename)
            filesize = os.path.getsize(filename)

            head_dic = {'filename': filename, 'filesize': filesize}
            head_raw = json.dumps(head_dic)
            head_len_struct = struct.pack('i', len(head_raw))
            conn.sendall(head_len_struct)
            conn.sendall(head_raw.encode('utf-8'))

            with open(filename, 'rb') as fp:
                data = fp.read()

```

3 服务器文件和数据库的维护：

服务器端（Server 文件夹下）存放有 txt 文件及相应的.jpg 格式封面，以及一个存储图书信息的 BOOKS.db 数据库（其中有一个 book_info 的表，存放图书名称以及阅读到多少页等信息），运行 run_server.py 启动服务器，服务器等待连接。

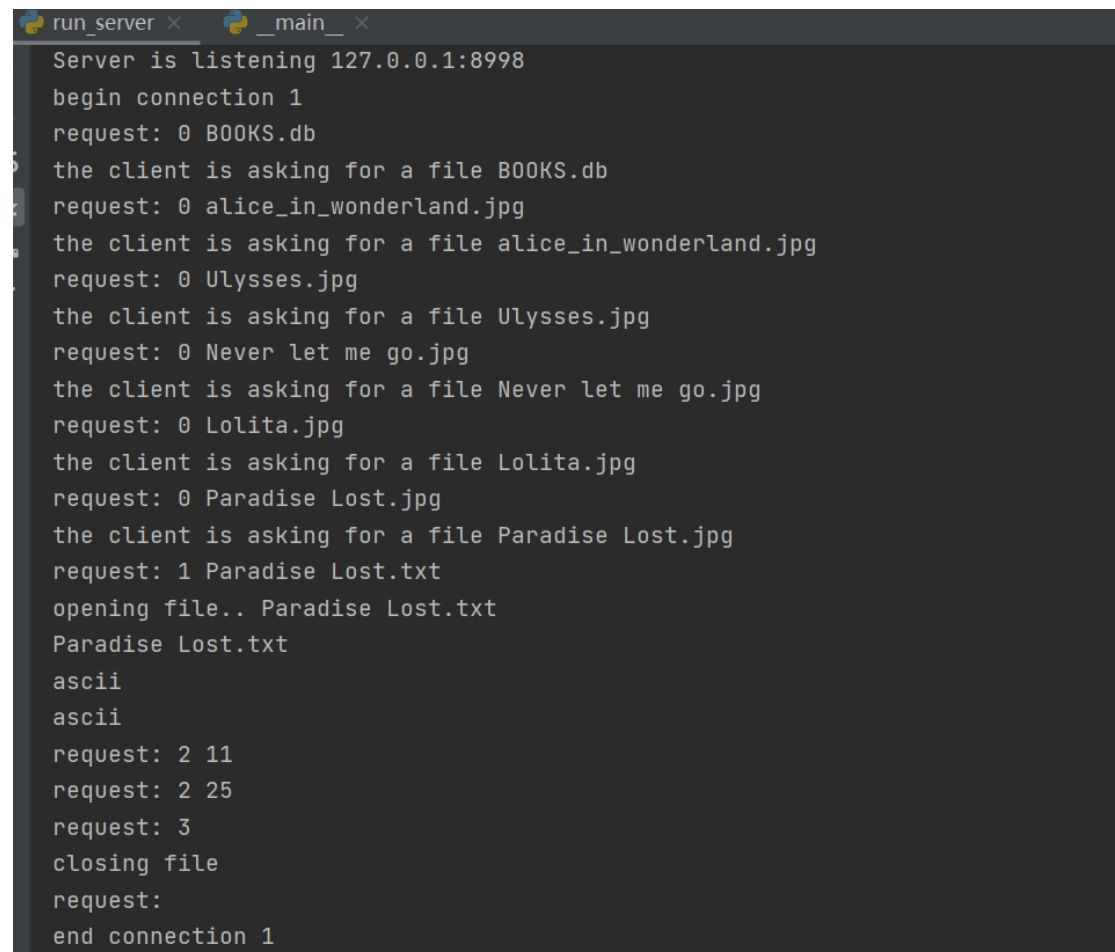
4 然后运行 Client/_main_.py 启动客户端（阅读器），阅读器启动时，首先向客户端请求下载 BOOKS.db 数据库（发送请求 0 BOOKS.db），然后根据数据库内容向服务器请求下载封面（发送请求 0 .jpg）并显示（如果有缓存则不需要下载），最后显示在前端窗口（书架）中。

前端书架是用 Qt Designer 中的 MainWindow 设计，实现了一个简单的封面维护功能，具体见 BookListWindow.py。

然后服务器继续等待客户端的行为，右击封面选择开始阅读，如阅读 Paradise Lost 客户端发送请求（1 Paradise Lost.txt），弹出阅读器窗口（阅读器的图形界面设计见 Client/TxtReader.py），服务器根据 BOOKS.db 中的信息将上次阅读到

页的内容传给客户端，按 ctrl+Z 可以向回翻一页，按 ctrl+G 可以向后翻一页，按 ctrl+G 会弹出一个翻页的窗口，客户端选择要去的页码，可以看出以上操作都是基于请求（2 num）完成的。关闭阅读器窗口，发送请求 3，再关闭书架窗口，则客户端关闭，断开连接。该过程如下图：

服务器：



```
run_server x  _main_ x
Server is listening 127.0.0.1:8998
begin connection 1
request: 0 BOOKS.db
the client is asking for a file BOOKS.db
request: 0 alice_in_wonderland.jpg
the client is asking for a file alice_in_wonderland.jpg
request: 0 Ulysses.jpg
the client is asking for a file Ulysses.jpg
request: 0 Never let me go.jpg
the client is asking for a file Never let me go.jpg
request: 0 Lolita.jpg
the client is asking for a file Lolita.jpg
request: 0 Paradise Lost.jpg
the client is asking for a file Paradise Lost.jpg
request: 1 Paradise Lost.txt
opening file.. Paradise Lost.txt
Paradise Lost.txt
ascii
ascii
request: 2 11
request: 2 25
request: 3
closing file
request:
end connection 1
```

客户端：

```
run_server x _main_ x
C:\Users\87670\AppData\Local\Programs\Python\Python38\python.exe D:/work/python/networkpj/Client/___main___.py
[Client] asking for book information.....
[TCP] loading... BOOKS.db, save as BOOKS_1.db
[TCP] length: 8192 , size: 8192 , time0.000997781753540039seconds
[Client] successfully update book information
[Client] asking for book covers.....
[TCP] loading... alice_in_wonderland.jpg
[TCP] length: 111589 , size: 111589 , time0.0019941329956054688seconds
[TCP] loading... Ulysses.jpg
[TCP] length: 16550 , size: 16550 , time0.0009698867797851562seconds
[TCP] loading... Never let me go.jpg
[TCP] length: 60013 , size: 60013 , time0.0009961128234863281seconds
[TCP] loading... Lolita.jpg
[TCP] length: 6987 , size: 6987 , time0.000997304916381836seconds
[TCP] loading... Paradise Lost.jpg
[TCP] length: 227170 , size: 227170 , time0.000997304916381836seconds
[Client] successfully download book covers
[Reader]exit txt reader
[Reader] file closed
```

具体的操作已在实验课上展示，这里不再赘述。

总结：

在这次项目中，我实现了一个简易的 txt 阅读器，支持了打开文本，翻页，跳页，下载封面和数据库，关闭等功能。总体来说，掌握了 socket 编程和前端设计的相关方法。不过该项目有许多可改进之处，比如虽然可以支持多个线程，但所有客户端都使用同一个数据库，可能需要增加用户登录功能加以改进。