**武汉大学计算机学院**

**本科生实验报告**

**网络及分布式计算实验**

专 业 名 称 ：软件工程

课 程 名 称 ：网络及分布式计算实验

指 导 教 师 ：胡继承 教授

团 队 成 员 一：刘瑞康（2016302580242）

团 队 成 员 二：朱 申（2016302580074）

二○一八年六月

**郑 重 声 明**

本团队呈交的实验报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告不包含他人享有著作权的内容。对本实验报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。

团队成员签名： 日期：2019-6-10

摘 要

本实验是网络及分布式计算的实验课程。目的是通过实践，掌握网络系统软件与应用软件开发的方法，能将所学的操作系统、数据库、软件工程、计算机网络等方面的知识集成到一起，开发实际软件系统。

实验参照FTP中文规范文档，通过C#提供的Socket（TCP/IP）编程组件进行FTP编程。使用WinForm以及IrisSkin4第三方库，进行界面设计和实现。同时，使用C#提供的Windows文件接口进行下载/上传的本地支持。

实现了一个能够进行简单的文件传输和自服务器所提供的根目录进行FTP服务器内目录浏览的简易FTP客户端。支持断点续传。拥有LOG功能。系统较为稳健，考虑了来自服务器的各种情况，支持来自服务器指定的多种编码格式。

**关键词：**FTP；C#；

**目 录**

**[1实验目的和意义 - 4 -](#_Toc632176799)**

[1.1 实验目的 - 4 -](#_Toc1393859084)

[1.2实验意义 - 4 -](#_Toc1838003312)

**[2 实验设计 - 5 -](#_Toc1916886336)**

[2.1 概述 - 5 -](#_Toc558976858)

[2.2 实验原理 - 5 -](#_Toc1630580428)

[2.3 实验方案 - 5 -](#_Toc1126434029)

**[3结论 - 7 -](#_Toc1908377098)**

[3.1 程序主要界面及结果 - 7 -](#_Toc1425618141)

3.2 程序 -18-

**[4 成员分工 - 25 -](#_Toc8402030)**

**[5 参考文献 - 25 -](#_Toc8402030)**

**1实验目的和意义**

**1.1 实验目的**

通过实践，掌握网络系统软件与应用软件开发的方法，能将所学的操作系统、数据库、软件工程、计算机网络等方面的知识集成到一起，开发实际软件系统。

**1.2实验意义**

通过实践，掌握网络系统软件与应用软件开发的方法，能将所学的操作系统、数据库、软件工程、计算机网络等方面的知识集成到一起，开发实际软件系统。

**2 实验设计**

**2.1 概述**

实现了一个能够进行简单的文件传输和自服务器所提供的根目录进行FTP服务器内目录浏览的简易FTP客户端。支持断点续传。拥有LOG功能。系统较为稳健，考虑了来自服务器的各种情况，支持来自服务器指定的多种编码格式。

**2.2 实验原理**

参照FTP中文规范文档，通过C#提供的Socket（TCP/IP）编程组件进行FTP编程。同时，使用两款本地的FTP服务器进行测试。使用WinForm以及IrisSkin4第三方库，进行界面设计和实现。同时，使用C#提供的Windows文件接口进行下载/上传的本地支持。

**2.3 实验方案**

1. 参阅FTP中文规范文档，根据要求进行功能设计。

2. 确定的功能实现需求是：实现简单的上传、下载功能同时都能支持断点续传。同时，为了调试和使用的方便，加入日志功能。

3. 根据2.中的要求功能和1.中的FTP中文规范文档，找出前端应该提供的信息 —— 即使用者应该提供的信息。包括：服务器IP、服务器端口、连接所用用户名、连接所用密码、本地文件上传的选择浏览框、目标服务器文件下载的浏览选择框、日志框以及暂停、中止命令、本地文件和目标服务器文件的手动刷新输入按钮。

4. 根据3.中得到的内容，配合实际使用方便程度设计出界面，包括以上的内容，同时加入当前任务指示栏、进度条、提供密码可见功能等。

5. 根据2.中的需求、1.中的规范、C#提供的Socket编程接口设计出功能实现的框架 ——

5.1. 封装从TCP/IP到FTP的编程接口，提供一个可以快捷实现FTP功能，外部可以直接进行功能调用，而不需要写重复的代码。实现高内聚低耦合。

5.2. 因为目的之一是实现断点续传，所以在进行上传/下载任务时必须分出另一个线程进行执行。所以，要封装一个任务类，中心控制类可以直接创建一个新的任务，让其执行。同时，中心控制类和任务类都可以调用封装好的FTP类。

5.3. 封装所有命令字符常量：因为这些命令都很有可能被多次使用，而且，命令作为字符串很容易出现错漏，因此归一到同一个地方管理很方便。

5.4. 同5.3.的理由，封装所有的提示或者可变的字符串。

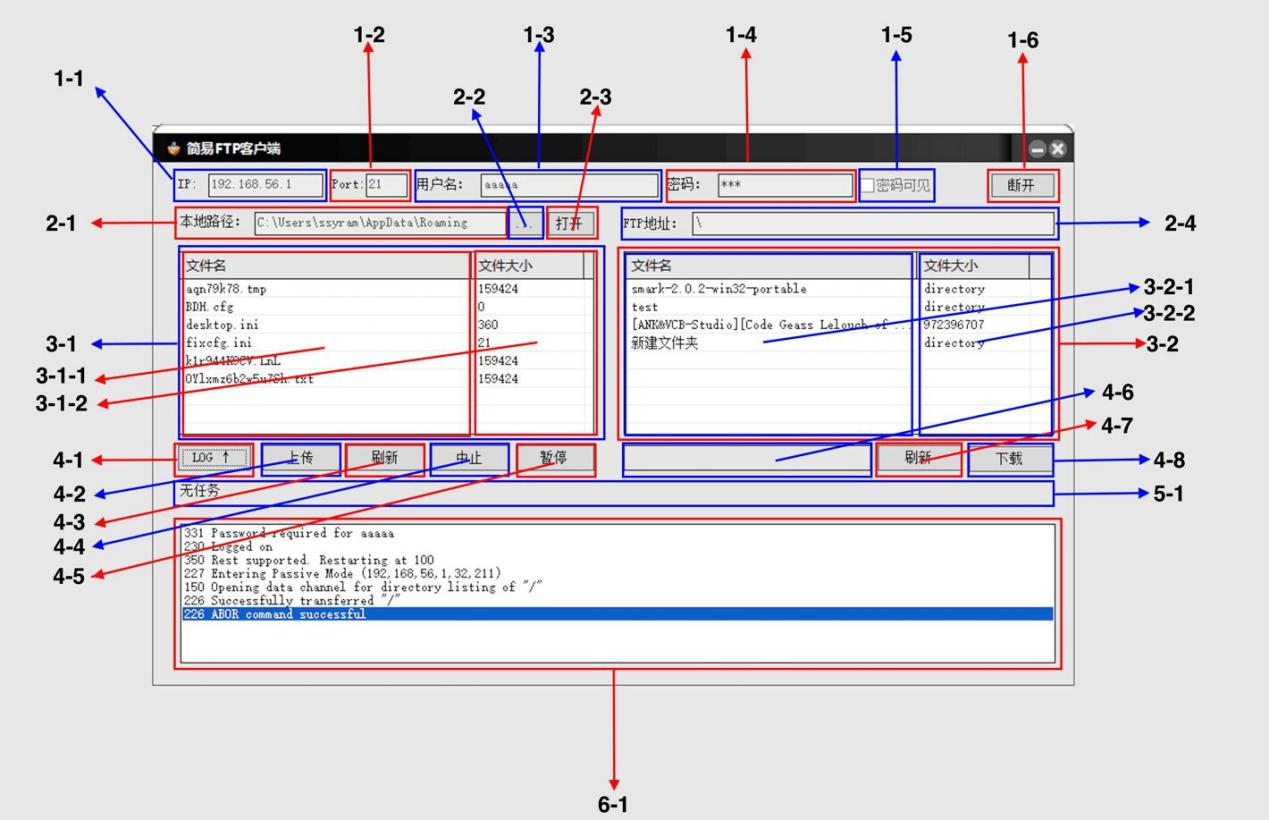
6. 根据框架设计进行实现，在实现中进一步完善各种功能。

7. 测试，提高系统稳健性。

8. 和根据实际使用修正界面设计，加入了可变动的LOG栏目，加入了可以直接通过Windows自带的文件浏览器（Explorer）打开当前的本地目录的功能。

**3结论**

**3.1 程序主要界面及结果**



程序界面介绍：

* 1. IP地址栏：这一栏支持的地址有IPv4和IPv6两种格式，点击连接按钮（当前1-6的断开按钮在没有连接的时候显示为连接）的时候会进行检查，如果IP地址格式不为正确的IPv4或者IPv6格式的话将会弹出提示框，并不予连接。
  2. 端口输入框：这个框有输入检查，不能输入非数字的内容，且当内部的数字大于65535时会自动变为65535，如果不输入直接按连接也会直接自动填充21。
  3. 登陆用用户名输入框
  4. 登陆密码输入框
  5. 密码可见选择框，选择以后密码可见。
  6. 连接/断开按钮：在连接之前表示为连接，在连接以后表示为断开。在传输或者命令正在执行的时候不允许断开。
  7. 本地路径显示框：一般来说会自动初始化到当前用户的应用数据的存放处，是个一直都是只读的窗口，要改变只能通过2-2.的按钮来改变，这一栏不能为空。
  8. 本地路径选择器：通过Windows的路径选择器来选择所希望的本地路径来改变2-1.的只读框的内容。
  9. 本地路径打开按钮：用Windows的文件浏览器（Explorer）来打开2-1.本地路显示框中的路径。
  10. FTP路径显示框：连接以后自动变为根地址，可以通过重新连接或者双击打开3-2.选择框中的目录文件来改变。
  11. 本地文件浏览框：浏览本地的对应目录的文件，需要注意的是只有文件，因为目录是可以通过2-2.本地路径选择器来进行选择的。可以通过双击项目或者选中某个项目以后点击4-2.上传按钮进行上传。

3-1-1. 本地文件浏览框的文件名栏目：用本地的Encoding来编码。

3-1-2. 本地文件浏览框的文件大小栏：单位是Byte（字节）。

* 1. FTP文件浏览框：浏览FTP对应目录下的所有文件及其大小的浏览框，这里除了文件以外还有目录，其中目录会在文件大小中表示为directory，双击会进入这个目录，改变FTP的路径而不是下载文件。同时选择的是一个目录项目的时候，4-8.下载按钮将和没有选中任何项目时候一样，点击没有反应。

3-2-1. FTP文件浏览框的文件名栏目：用服务器的编码方式进行编码。

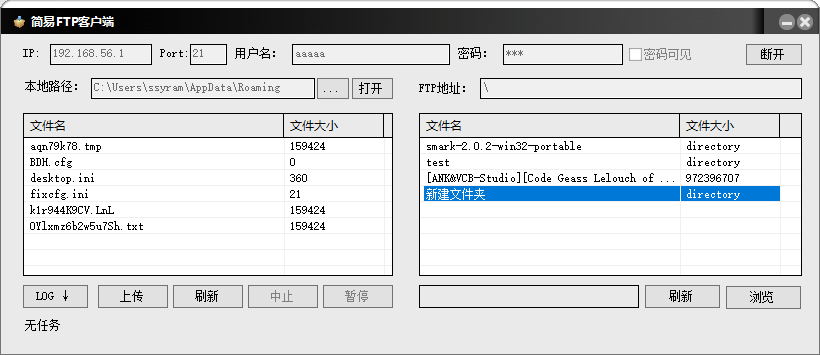
3-2-2. FTP文件浏览框的文件大小栏：如果是文件，单位是Byte（字节）。

* 1. LOG显示/隐藏按钮：可以显示/隐藏下面的6-1.日志显示框。
  2. 本地文件上传按钮：当选中3-1.本地文件选择框中的文件的时候点击这个按钮将会上传文件，如果没有选中，点击将不会有反应。
  3. 本地文件浏览框刷新按钮：刷新3-1.本地文件浏览框列表信息。
  4. 中止传输按钮：只有在有传输任务的时候才可以点击，可以立刻中止当前的传输任务。
  5. 暂停/继续传输按钮：只有在有任务传输的时候才可以点击，可以中止当前的传输，然后这个按钮将会变为继续传输按钮，点击继续传输可以继续没有传输完的任务。注意这个功能需要看服务器端的支持程度，部分服务器并不支持这个功能。如果不支持将会报错。
  6. 传输进度条：可以直观感受当前的传输进度。
  7. FTP目录浏览框刷新按钮：可以刷新3-2.FTP目录浏览框的列表信息，如果发现已经断开了连接，将会断开连接。
  8. 下载/浏览按钮：如果选中的是3-2.FTP目录浏览框中的文件项目，将会是下载按钮，点击的效果是将内容下载到本地；如果选中的是3-2.FTP目录浏览框中的目录项目，将会是浏览按钮，点击的效果是进入这个目录进行浏览。
  9. 当前任务显示条：显示当前正在进行的任务。
  10. 日志框：在这里会逐条显示日志。



启动界面说明：

1. 启动的时候默认会打开本地路径里面的一般用户存储应用数据的地方，同时加载内容到本地浏览框，但是都是不能进行操作的，只能用Windows的文件浏览器浏览这个目录来调整内部的内容。
2. 端口在初始时候就是21。
3. 启动的时候下方的功能键除了“打开”、“LOG”以外都是不能使用的。



选中FTP目录浏览框中的目录项目以后，下载按钮将会变成浏览按钮。



密码可见框演示。



在连接时，输入了不正确格式的IP地址会报错。

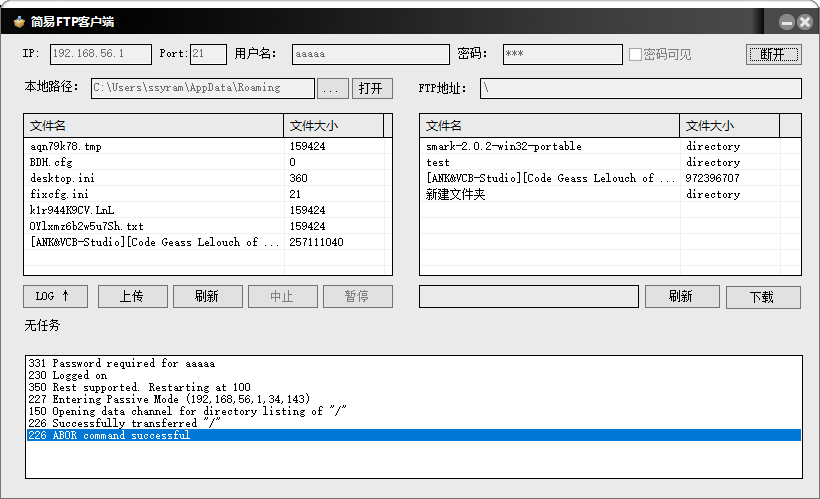


端口输入框不能输入除了数字之外的内容，同时，如果输入的数字大于65535，将会自动变为65535。

如果不输入端口，在连接时也会自动补充为21。



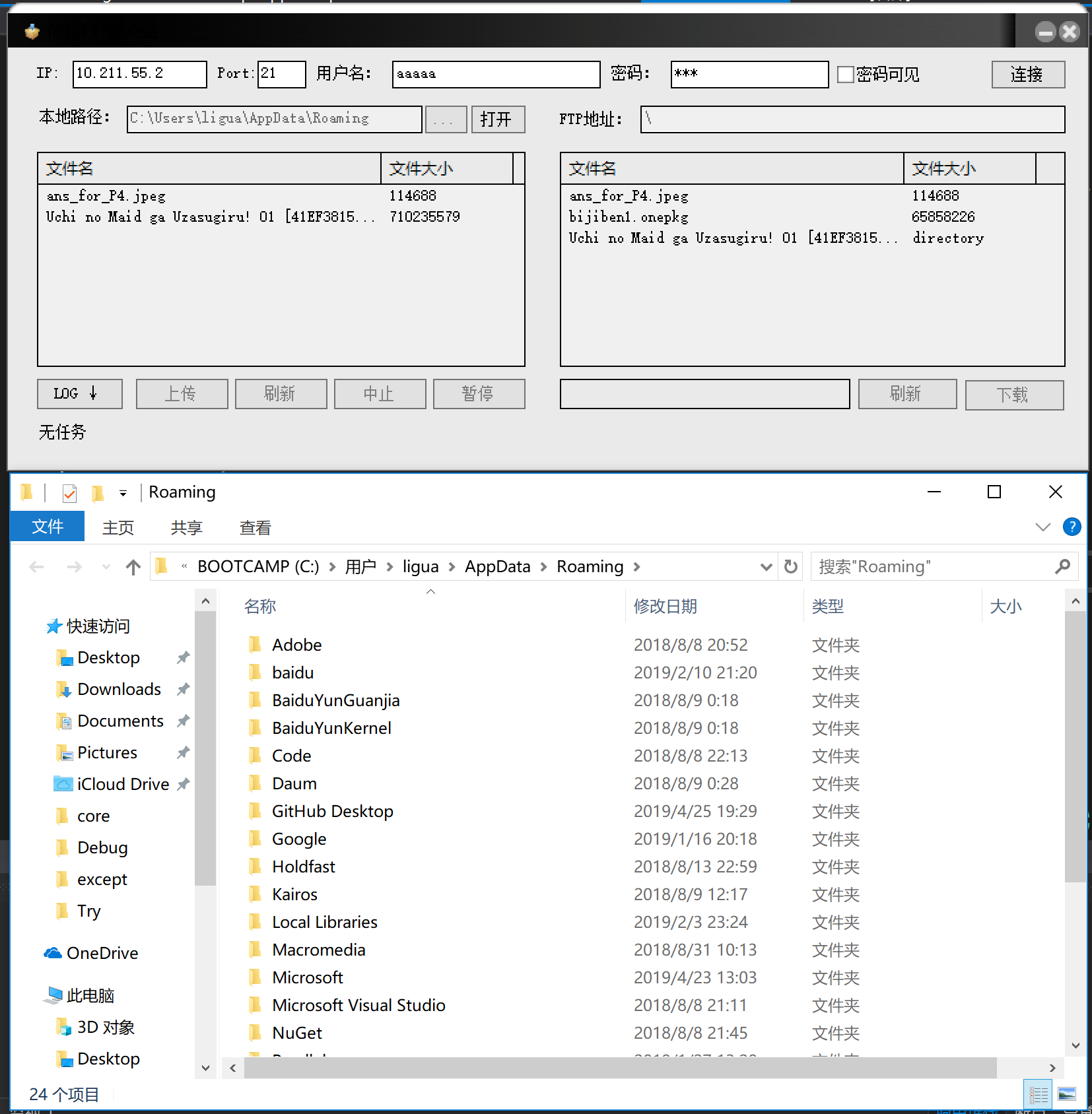
如果在连接过程中出现连接不到服务器的错误，会在点击连接按钮之后贴示“连接超时或者错误”。



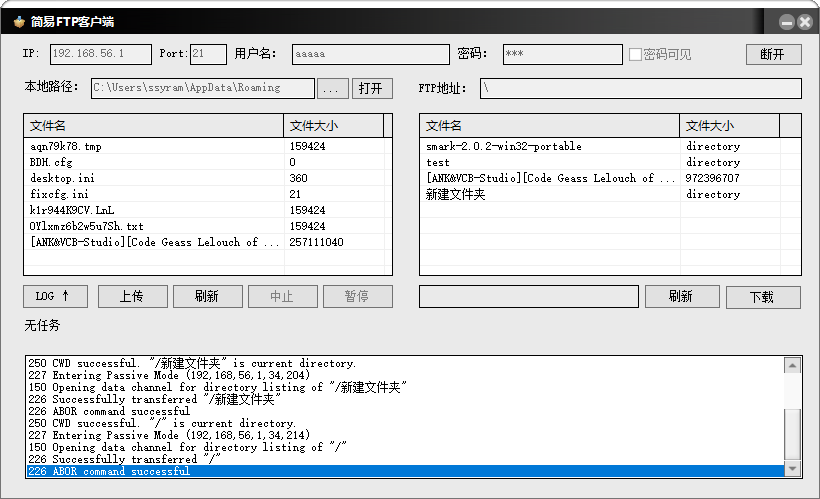
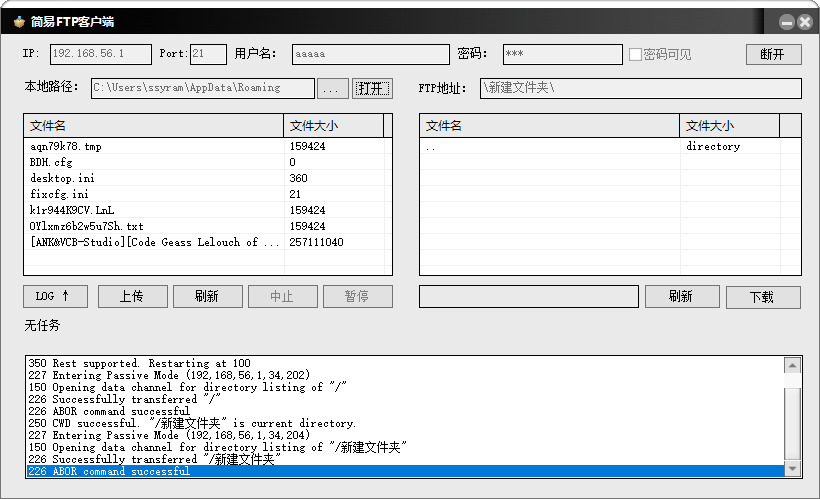
连接后的界面：

1. 支持多种服务器端的中文编码。
2. 上面的输入栏不能再修改，除非断开连接。

点击“LOG↑”或者“LOG↓”可以将信息打印窗口合上或者展开。

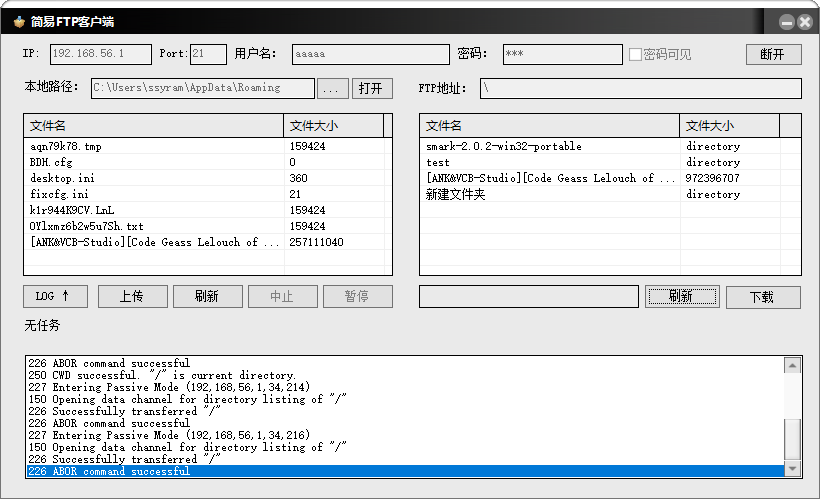


点击本地路径右侧的打开按钮，可以打开相应的本地路径，用来查看本地文件夹。这样可以检查下载的文件是否正确地放到想要的位置。如上图所示。

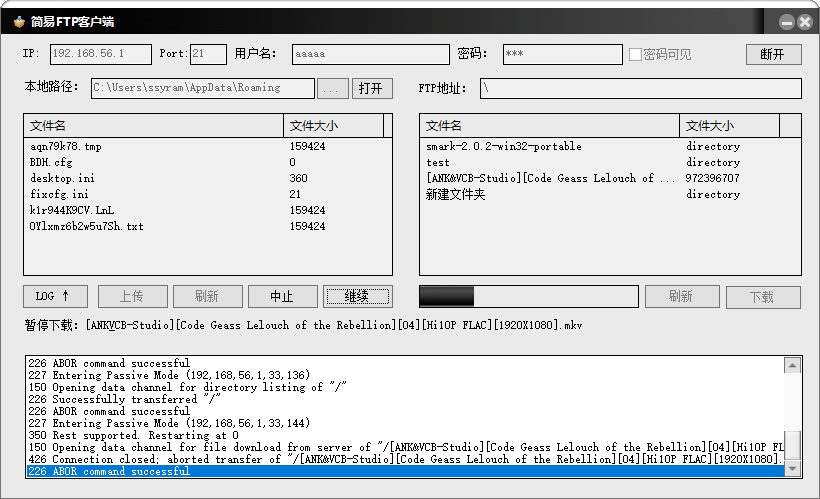


双击一个文件大小为directory的项目（服务器端文件夹）或者选中它以后点击浏览按钮可以进入这个目录。

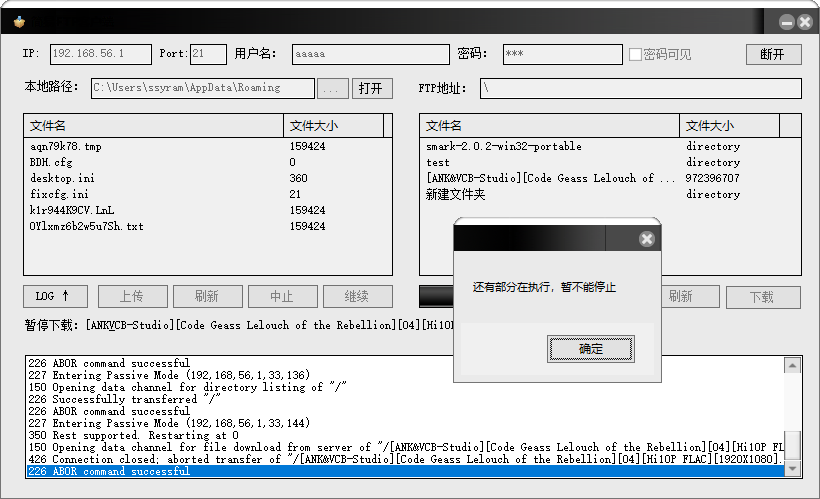
双击栏目最上方的“..”，或者选中它以后点击浏览按钮可以回归到上层目录。



通过点击刷新可以刷新界面：从下方的LOG中可以看出进行了刷新。

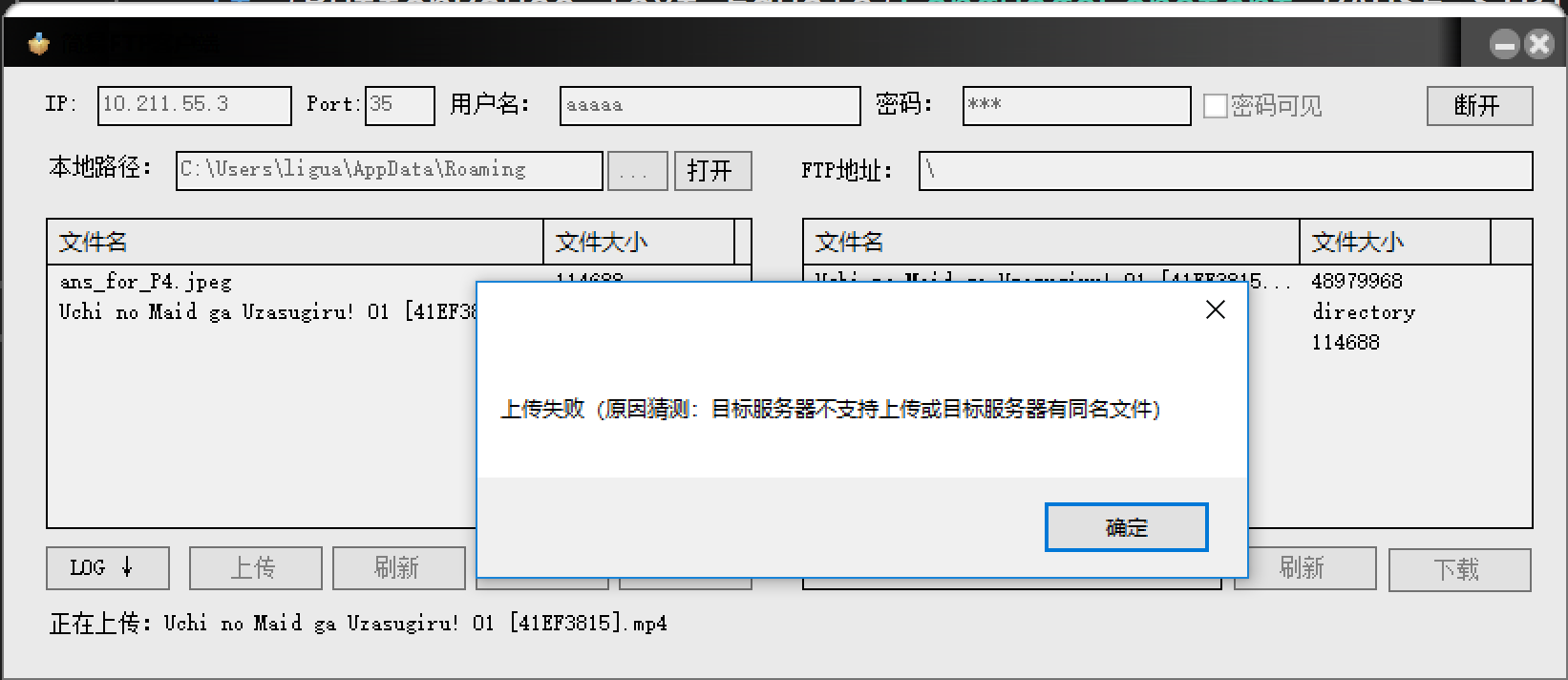


断点续传实例。



断点续传的时候不能断开连接。

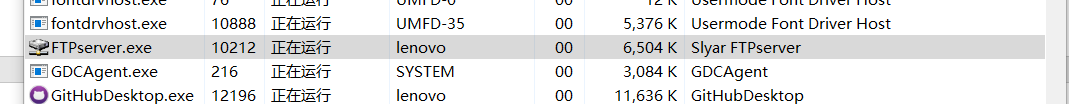
如果上传的文件的文件名与所上传文件夹中的某个文件的文件名相同，那么程序会进行相应的提示。



程序健壮性保证：



当正常连接时，因服务器出现故障或网络原因导致异常断开，本地客户端会检测出TCP连接已断开，后续的用户操作如下载、刷新和断开都会保证被客户端正常处理，并不抛异常。



通过在任务管理器中结束FTP服务进程模拟服务器崩溃或网络异常等情况。

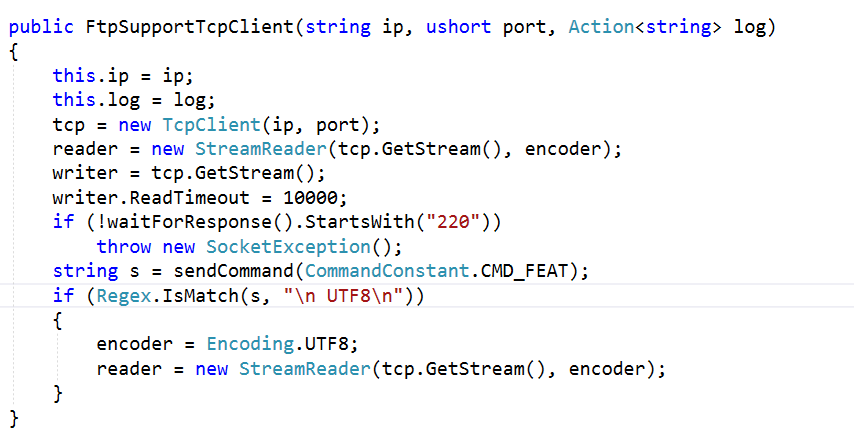


这时再点击刷新或者下载，服务器目录会被清空，点击断开即可正常退出。

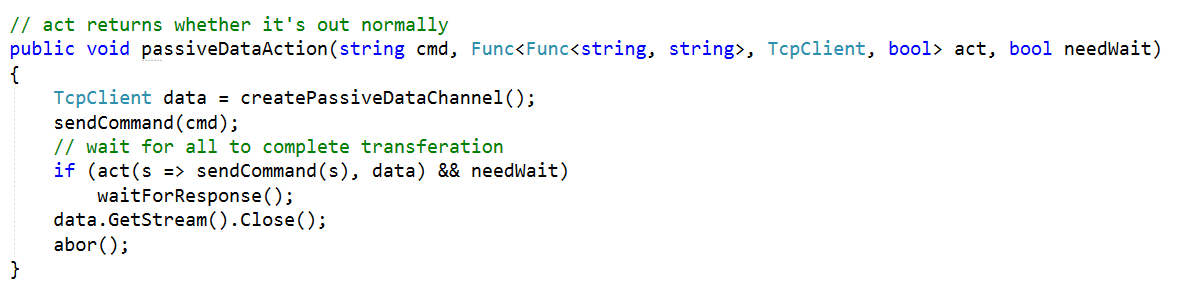
**3.2 程序**

* 用Socket实现的FTP的类

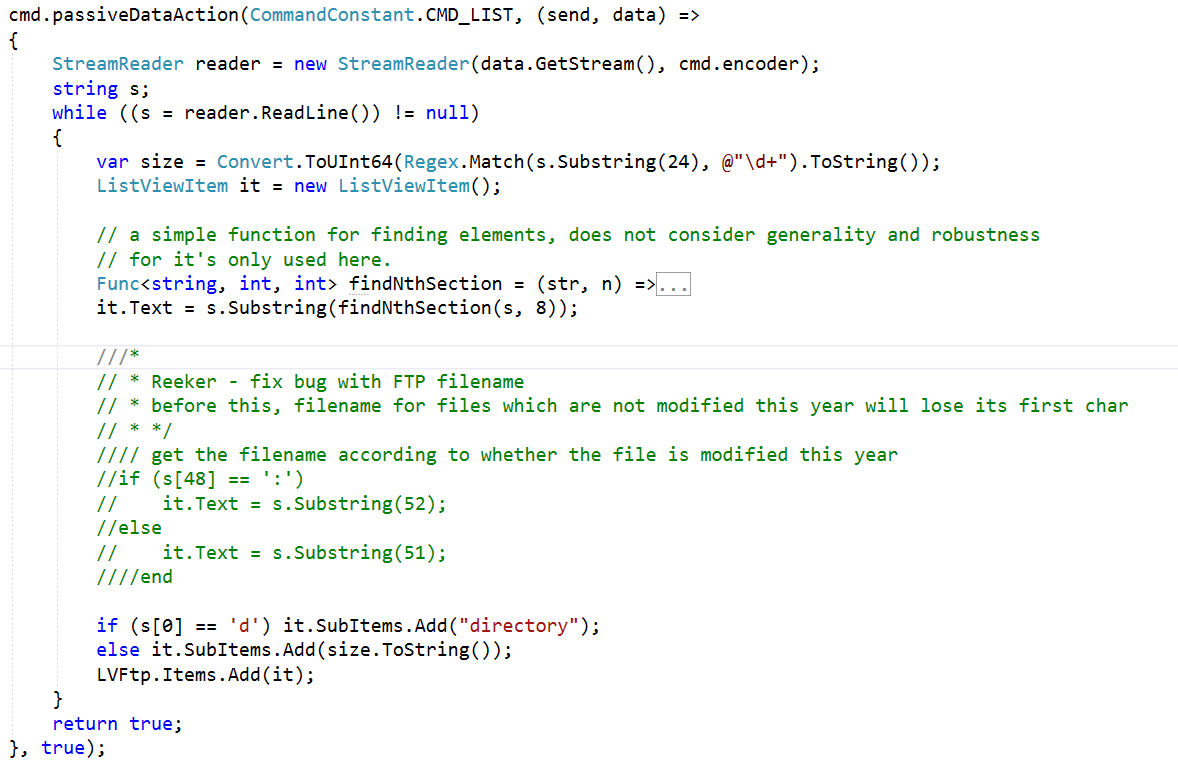
传入对应IP和端口，建立对应的连接，并且判断是否拥有这样的编码：



* 对数据传输等的操作使用传入函数进行计算的方式，将数据传输流的获取被动连接并且此后的关闭都封装在其中。



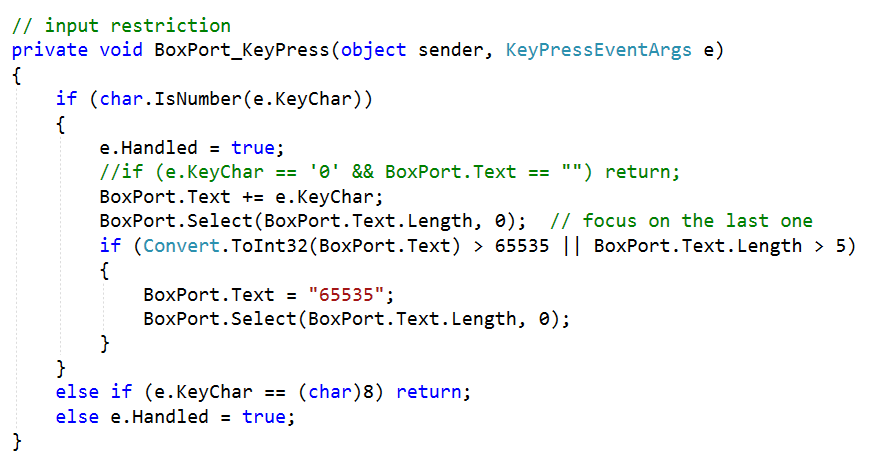
使用的实例：



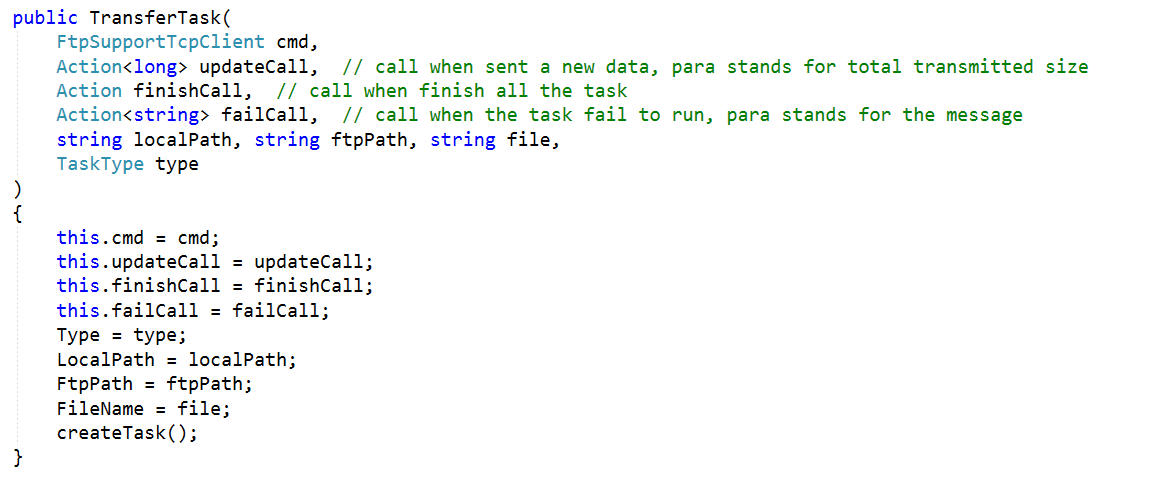
这是实现FTP目录浏览框的实现代码。只需要传入一个包装好的执行函数即可，执行函数接受一个发送的方式函数，可以通过这个函数发送命令，同时获得了数据通道的Socket（data）。 其返回值表示是否属于正常退出，是否需要等待最后一条命令。

* 端口输入限制

限制只能输入数字和输入的数字范围。

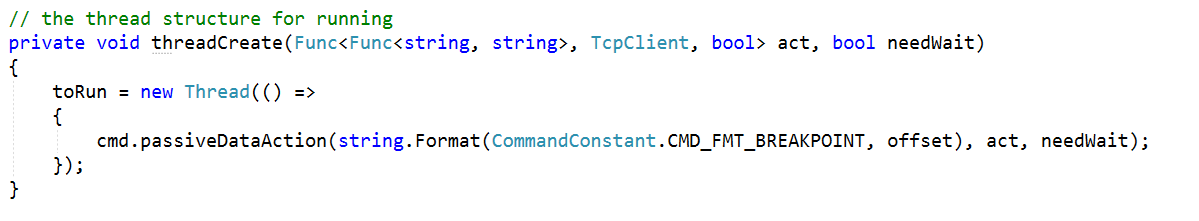


* 断点续传的实现
  + 首先设计任务类的设计，因为任务是分出线程实现的，同时，为了保持和主任务类的不相互干涉，降低耦合度 —— 控制住操控一个类的方式只在这个类内部执行，不涉及对某个类内部实现的猜测 —— 这种猜测将会大大增加方案修改的成本，事实是，如果直接获取内部类的内容指针，然后直接调用修改UI，将会极大增加新组件添加的时间成本。因此采用传入操作函数的方式 —— 外部指定回调函数。



* + 创建线程和实现断点续传的关键：

线程创建以后将会首先发送下一个期待的断点，然后发送所希望传输的文件的指令。

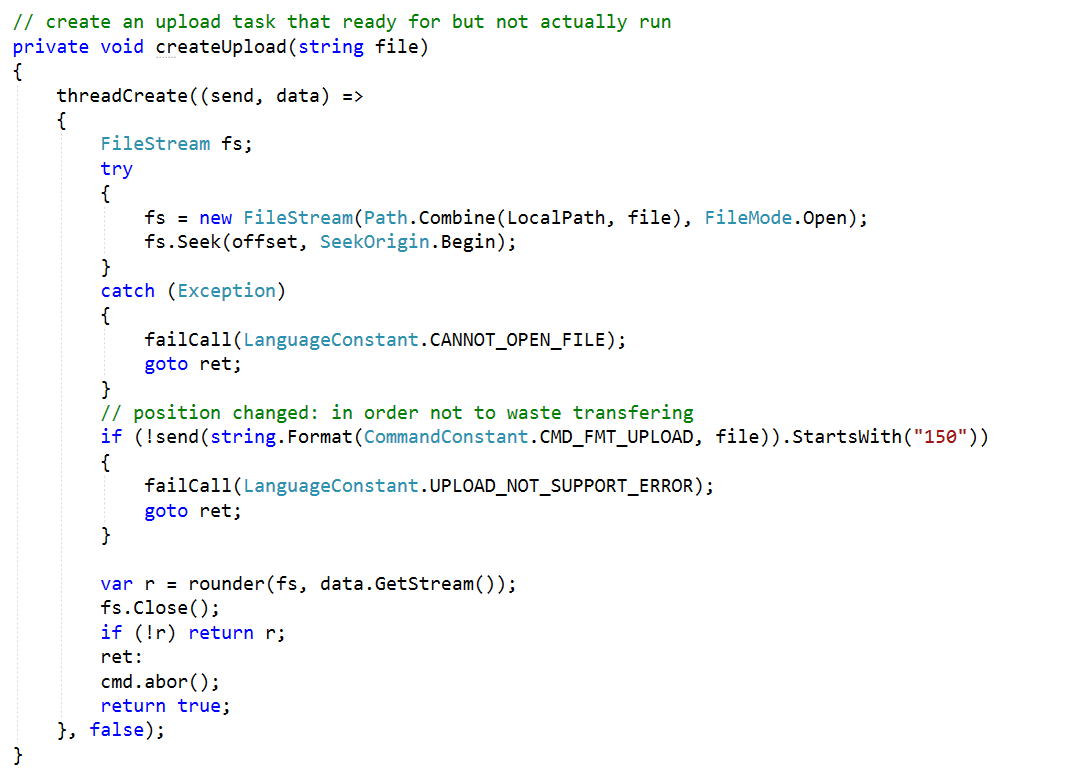


通过REST命令执行断点续传功能：



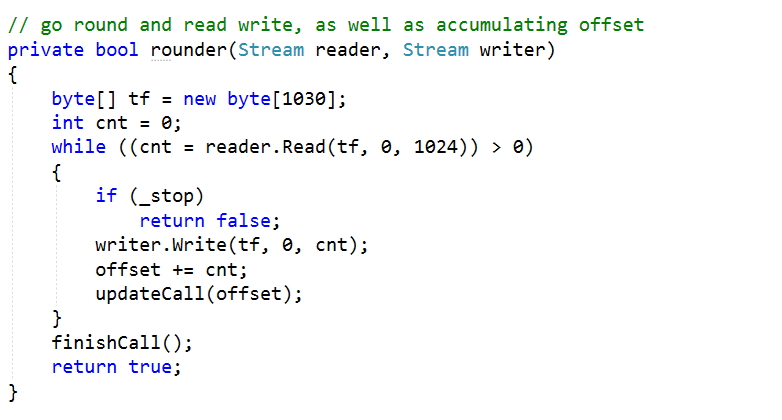
同时act传入上传/下载的内容函数。

执行上传和下载的代码：

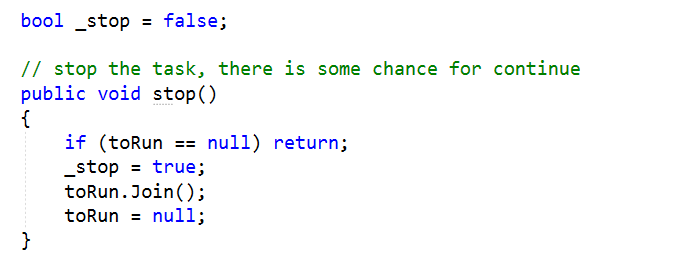




其中因为两者都是有相同的执行逻辑，因此将这个执行逻辑进行了抽象，变成了一个rounder，执行的操作是将内容从Reader读取写入Writer，同时检测如果停止指令为真，则返回非正常退出的信号，通知FtpSupportTcpClient::passiveDataAction不需要等待。



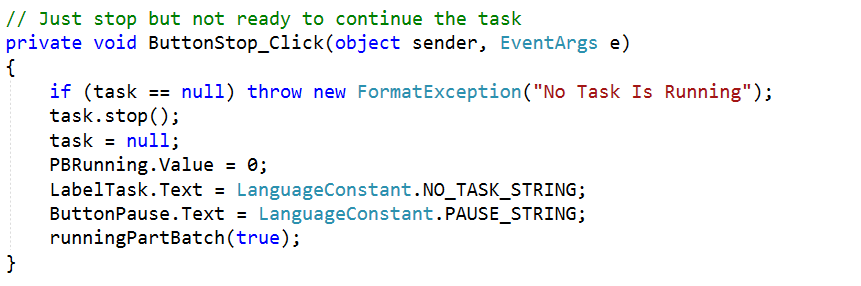
对于一个特定的任务来说，不存在所谓暂停的说法，因此这里是真正的退出任务，外部调用退出的函数的内部执行：



这里首先发出暂停的命令，然后等待线程回归后才退出。

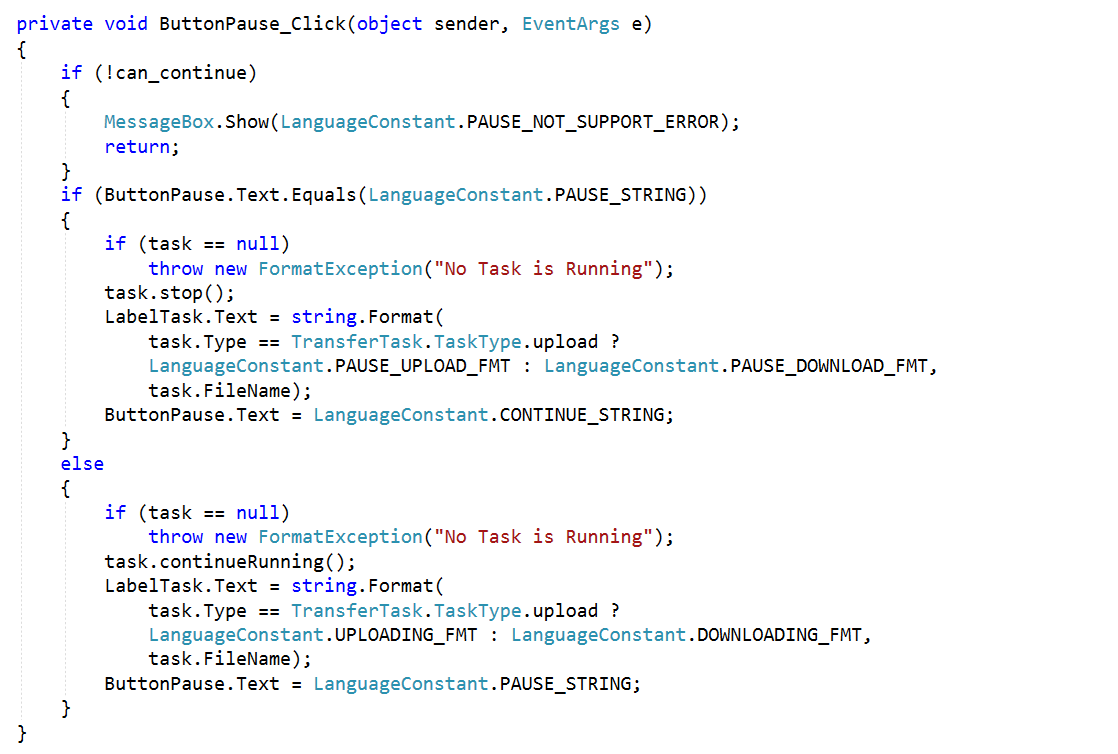
是否在外部视角的退出 —— 直观来说是按了中止按钮或者暂停按钮的区别，是这个任务对象是否被抛弃 —— 对应代码：

中止代码：



可以看出这里抛弃了整个任务。

暂停代码：

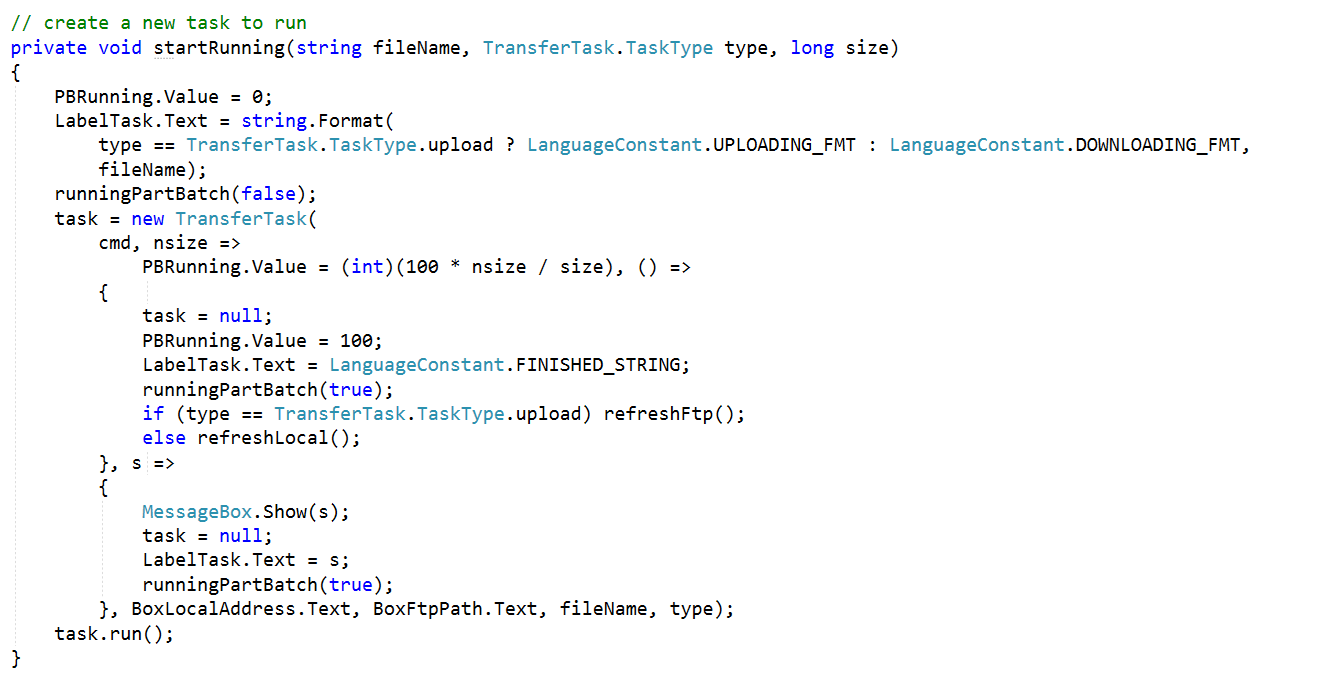


这里首先检测一下对象服务器是否支持断点续传，如果不支持，将会不予暂停。

暂停的地方可以看出只是暂停，没有抛弃任务对象。

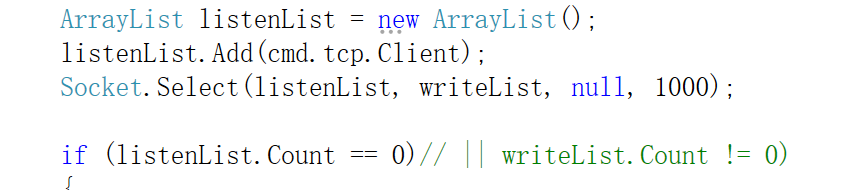
等待按下继续按钮或者按下中止按钮，前者是继续任务传输 —— 从rounder中保持下来的断点信息将会被利用实现断点续传。后者是直接抛弃任务对象。

* + 在外部的任务创建



可以看出在执行过程中，将会不断更新进度条。任务正常完成以后将会抛弃任务对象。

如果发生了一些内部的错误，将会将信息通过弹窗提示框的方式表现出来，同时抛弃任务对象。



* + 通过监听socket来保证连接仍然成功，只有连接成功，才会进行后续的动作，保证了程序的健壮性。

**4 成员分工**

2016302580242-刘瑞康：主任务类实现、断点续传实现、包装FTP支持的TCP类、传输任务类的实现。使用第三方库对窗体外观进行整体美化，程序外观属性配置。

2016302580074-朱申：界面设计。维护程序健壮性，在服务端异常、客户端异常和网络环境异常等情况下保证程序能够正常运行并退出。优化了程序的部分逻辑。对于程序中可能出现的错误进行处理，搭建测试环境，并对最终程序进行了测试并修复出现的bug。

**5 参考文献**

[1] 《网络编程实用教程》（叶树华等编著，人民邮电出版社）

|  |
| --- |
| 【结论】：  实现了一个能够进行简单的文件传输和自服务器所提供的根目录进行FTP服务器内目录浏览的简易FTP客户端。支持断点续传。拥有LOG功能。系统较为稳健，考虑了来自服务器的各种情况，支持来自服务器指定的多种编码格式。 |
| 【小结】：  该实验是理论知识和动手能力的综合体现。通过本实验，掌握了网络系统软件、网络应用软件的开发方法、开发平台的使用、与实际数据库的集成方法。  实验参照了FTP中文规范文档，通过C#提供的Socket（TCP/IP）编程组件进行FTP编程。同时，使用两款本地的FTP服务器进行测试。使用WinForm以及IrisSkin4第三方库，进行界面设计和实现。同时，使用C#提供的Windows文件接口进行下载/上传的本地支持。 |
| 指导老师评语及成绩 |
| 【评语】：  成 绩： 指导老师签名：  批阅日期： |