ftplib- FTP 客户端

版本：1.0

日期：2026-01-09

状态：受控/临时文件

文件保密级别：（请勾选 ■ ）		
绝密 □	保密 □	公开 ■

文件管控表

文件变更记录			
修订日期	最新版本	修订内容	编制
2026-01-09	1	首次发行	Wells Li

上海移远通信技术股份有限公司（以下简称“移远通信”）始终以为客户提供最及时、最全面的服务为宗旨。如需任何帮助，请随时联系我司上海总部，联系方式如下：

上海移远通信技术股份有限公司
上海市闵行区田林路 1016 号科技绿洲 3 期（B 区）5 号楼　　邮编：200233
电话：+86 21 5108 6236　　邮箱：info@quectel.com

或联系我司当地办事处，详情请登录：http://www.quectel.com/cn/support/sales.htm。

如需技术支持或反馈我司技术文档中的问题，请随时登录网址：
http://www.quectel.com/cn/support/technical.htm 或发送邮件至：support@quectel.com。

前言

移远通信提供该文档内容以支持客户的产品设计。客户须按照文档中提供的规范、参数来设计产品。同时，您理解并同意，移远通信提供的参考设计仅作为示例。您同意在设计您目标产品时使用您独立的分析、评估和判断。在使用本文档所指导的任何硬软件或服务之前，请仔细阅读本声明。您在此承认并同意，尽管移远通信采取了商业范围内的合理努力来提供尽可能好的体验，但本文档和其所涉及服务是在“可用”基础上提供给您的。移远通信可在未事先通知的情况下，自行决定随时增加、修改或重述本文档。

使用和披露限制

许可协议

除非移远通信特别授权，否则我司所提供硬软件、材料和文档的接收方须对接收的内容保密，不得将其用于除本项目的实施与开展以外的任何其他目的。

版权声明

移远通信产品和本协议项下的第三方产品可能包含受移远通信或第三方材料、硬软件和文档版权保护的相关资料。除非事先得到书面同意，否则您不得获取、使用、向第三方披露我司所提供的文档和信息，或对此类受版权保护的资料进行复制、转载、抄袭、出版、展示、翻译、分发、合并、修改，或创造其衍生作品。移远通信或第三方对受版权保护的资料拥有专有权，不授予或转让任何专利、版权、商标或服务商标权的许可。为避免歧义，除了正常的非独家、免版税的产品使用许可，任何形式的购买都不可被视为授予许可。对于任何违反保密义务、未经授权使用或以其他非法形式恶意使用所述文档和信息的违法侵权行为，移远通信有权追究法律责任。

商标

除另行规定，本文档中的任何内容均不授予在广告、宣传或其他方面使用移远通信或第三方的任何商标、商号及名称，或其缩略语，或其仿冒品的权利。

第三方权利

您理解本文档可能涉及一个或多个属于第三方的硬软件和文档（“第三方材料”）。您对此类第三方材料的使用应受本文档的所有限制和义务约束。

移远通信针对第三方材料不做任何明示或暗示的保证或陈述，包括但不限于任何暗示或法定的适销性或特定用途的适用性、平静受益权、系统集成、信息准确性以及与许可技术或被许可人使用许可技术相关的不侵犯任何第三方知识产权的保证。本协议中的任何内容都不构成移远通信对任何移远通信产品或任何其他硬软件、设备、工具、信息或产品的开发、增强、修改、分销、营销、销售、提供销售或以其他方式维持生产的陈述或保证。此外，移远通信免除因交易过程、使用或贸易而产生的任何和所有保证。

隐私声明

为实现移远通信产品功能，特定设备数据将会上传至移远通信或第三方服务器（包括运营商、芯片供应商或您指定的服务器）。移远通信严格遵守相关法律法规，仅为实现产品功能之目的或在适用法律允许的情况下保留、使用、披露或以其他方式处理相关数据。当您与第三方进行数据交互前，请自行了解其隐私保护和数据安全政策。

免责声明

1) 移远通信不承担任何因未能遵守有关操作或设计规范而造成损害的责任。
2) 移远通信不承担因本文档中的任何因不准确、遗漏、或使用本文档中的信息而产生的任何责任。
3) 移远通信尽力确保开发中功能的完整性、准确性、及时性，但不排除上述功能错误或遗漏的可能。除非另有协议规定，否则移远通信对开发中功能的使用不做任何暗示或法定的保证。在适用法律允许的最大范围内，移远通信不对任何因使用开发中功能而遭受的损害承担责任，无论此类损害是否可以预见。
4) 移远通信对第三方网站及第三方资源的信息、内容、广告、商业报价、产品、服务和材料的可访问性、安全性、准确性、可用性、合法性和完整性不承担任何法律责任。

目录

1 模块概述

ftplib 模块用于向服务器发起 FTPS 请求，是一种用于在计算机网络中传输文件的标准协议，它基于客户端-服务器模型，通过控制连接和数据连接实现文件的上传、下载和管理操作。

2 构造函数

2.1. ftplib.FTP

2.1.1. 接口概述

构建 FTP 连接对象。

2.1.2. 函数原型

class ftplib.FTP(host=None, port=None, user=None, passwd=None, acct=None, timeout=None)

2.1.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
host	否	string	-	FTP 服务端地址，未传入时需自行调用 connect 方法连接服务端
port	否	int	21	FTP 服务端端口
user	否	string	anonymous	客户端登录用户名，若未指定则匿名访问
passwd	否	string	anonymous@	客户端登录密码，如果 user 为'anonymous'，那么默认的 passwd 是'anonymous@
acct	否	string	-	账户，一般不做设置
timeout	否	int	-	连接超时，单位为秒

- 实例化参数时如主动传入 host 与 port，将在对象构建成功后主动调用 connect 方法连接服务端，连

接失败会触发异常

- 在主动连接基础上传入 user 与 passwd 会在连接成功后调用 login 方法登录服务端

2.1.4. 返回值描述

返回 FTP 对象。

2.1.5. 示例

```
# 未抛出异常则包含该功能
from ftplib import FTP

# 不传入参数创建 FTP 对象
ftp = FTP()

# 传入参数创建 FTP 对象
host = '192.168.1.100'
port = 21
user = 'testuser'
password = 'test123'
ftp = FTP(host=host, port=port, user=user, passwd=password)
```

2.2. ftplibtls.FTP_TLS

2.2.1. 接口概述

构建 FTP_TLS 连接对象。

2.2.2. 函数原型

```
class
ftplibtls.FTP_TLS(host=None, port=None, user=None, passwd=None, acct=None, timeout=None, ssl_
params=None, implicit=True)
```

2.2.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
host	否	string	-	FTP 服务端地址，未传入时需自行调用 connect 方法连接服务端
port	否	int	implicit-990 explicit-21	FTP 服务端端口
user	否	string	anonymous	客户端登录用户名，若未指定则匿名访问
passwd	否	string	anonymous@	客户端登录密码，如果 user 为'anonymous'，那么默认的 passwd 是'anonymous@
acct	否	string	-	账户，一般不做设置

timeout	否	int	-	连接超时，单位为秒
ssl_params	否	dict	-	SSL 参数配置 ssl_params={ca: 'xxx', session_reuse: 'false '}
implicit	否	bool	True	SSL 连接模式 • True：隐式 SSL（端口 990） • False：显式 SSL（端口 21）

- 实例化参数时如主动传入 host 与 port，将在对象构建成功后主动调用 connect 方法连接服务端，连接失败会触发异常
- 在主动连接基础上传入 user 与 passwd 会在连接成功后调用 login 方法登录服务端
- FTPS 同一时刻只支持单个连接
- SSL 不支持双向认证

2.2.4. 返回值描述

返回 FTP_TLS 对象。

2.2.5. 示例

```
# 未抛出异常则包含该功能
from ftplibtls import FTP_TLS

# 不传入参数创建 FTP_TLS 对象
ssl_params = {
    "ca": ca_content,              # CA 证书内容
    "session_reuse": True,         # 控制通道会话是否复用
}

ftp = FTP_TLS(ssl_params=ssl_params)

# 传入参数创建 FTP_TLS 对象
def read_cert_file(filepath):
    with open(filepath, 'rb') as f:
        return f.read()
host = '192.168.1.100'
port = 990
user = 'testuser'
password = 'test123'
ca_content = read_cert_file('/flash/ftp_ca.pem')
ssl_params = {
    "ca": ca_content,              # CA 证书内容
    "session_reuse": True,         # 控制通道会话是否复用
}
ftp = FTP_TLS(host=host, port=port, user=user, passwd=password, ssl_params=ssl_params)
```

3 连接与配置

3.1. ftp.set_debuglevel

3.1.1. 接口概述

设置实例的调试级别。

3.1.2. 函数原型

ftp.set_debuglevel(level)

3.1.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
level	是	int	0	调试级别

- 0：不产生调试信息
- 1：产生中等数量的调试信息，通常每个请求产生一行
- ≥2：产生的调试信息最多，FTP 控制连接上发送和接收的每一行都将被记录

3.1.4. 返回值描述

无返回值。

3.1.5. 示例

ftp.set_debuglevel(1)

3.2. ftp.connect

3.2.1. 接口概述

FTP 客户端发起连接服务端请求。

3.2.2. 函数原型

ftp.connect(host, port, timeout)

3.2.3. 参数描述

参数名	是否必填	数据类型	默认值	说明

host	是	string	-	FTP 服务端地址
port	否	int	21	FTP 服务端端口
timeout	否	int	-	连接超时，单位为秒

3.2.4. 返回值描述

返回以 220 开头的字符串，则表示连接成功（如' 220 (vsFTPd 3.0.2) '），否则表示连接失败。

3.2.5. 示例

```
host = '192.168.1.100'
port = 21

result = ftp.connect(host=host, port=port)
print(result)   # 220 (vsFTPd 3.0.2)
```

3.3. ftp.login

3.3.1. 接口概述

FTP 客户端发起登录服务端请求，登录请求需在连接建立成功后进行。

3.3.2. 函数原型

```
ftp.login(user, passwd)
```

3.3.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
user	是	string	anonymous	客户端登录用户名
passwd	否	string	anonymous@	客户端登录密码

3.3.4. 返回值描述

返回以 230 开头的字符串表示登录成功（如'230 Login successful.'），否则表示登录失败。

3.3.5. 示例

```
user = 'testuser'
passwd = 'test123'

result = ftp.login(user=user, passwd=passwd)
print(result)   # 230 Login successful.
```

4 FTP 操作指令

4.1. ftp.nlst

4.1.1. 接口概述

返回文件名列表，默认返回当前服务器目录。

4.1.2. 函数原型

ftp.nlst(*args)

4.1.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
*args	否	可变参数	-	可传入一个目录路径参数，格式为字符串。不传入参数时列出当前服务器目录

4.1.4. 返回值描述

文件名列表。

4.1.5. 示例

```
files = ftp.nlst('/FTP-TEST ')
print(files)
```

4.2. ftp.dir

4.2.1. 接口概述

获取当前目录中的内容列表。

4.2.2. 函数原型

ftp.dir(*args, **kw)

4.2.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
*args	否	可变参数	-	可传入一个目录路径参数，格式为字符串。不传入参数时列出当前服务器目录

**kw	否	关键参数	-	支持的关键字参数:`callback`：回调函数，用于处理目录列表的每一行

4.2.4. 返回值描述

无返回值。

4.2.5. 示例

```
directory_list = []
def collect_lines(line):
    directory_list.append(line)

ftp.dir(callback=collect_lines)  # 传递回调函数
print("目录列表:")
for i, line in enumerate(directory_list, 1):
    print(f"{i:3d}: {line}")
```

4.3. ftp.pwd

4.3.1. 接口概述

获取当前 FTP 服务器目录。

4.3.2. 函数原型

```
ftp.pwd()
```

4.3.3. 参数描述

无。

4.3.4. 返回值描述

当前 FTP 服务器目录。

4.3.5. 示例

```
current_dir = ftp.pwd()
print(current_dir)
```

4.4. ftp.rename

4.4.1. 接口概述

重命名 FTP 服务器上的文件。

4.4.2. 函数原型

ftp.rename(fromname, toname)

4.4.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
fromname	是	str	-	待修改的文件名
toname	是	str	-	重命名后的文件名

4.4.4. 返回值描述

返回以 250 开头的字符串，表示重命名成功（如'250 Rename successful.'），否则表示重命名失败。

4.4.5. 示例

```
result = ftp.rename('test1.txt', 'test2.txt')
print(result)   # 250 Rename successful.
```

4.5. ftp.delete

4.5.1. 接口概述

删除 FTP 服务器上的文件。

4.5.2. 函数原型

ftp.delete(file_name)

4.5.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
file_name	是	str	-	待删除的文件名

4.5.4. 返回值描述

返回以 250 开头的字符串，表示删除成功（如'250 Delete operation successful.'），否则表示删除失败。

4.5.5. 示例

```
result = ftp.delete('test1.txt')
print(result)   # 250 Delete operation successful.
```

4.6. ftp.mkd

4.6.1. 接口概述

在 FTP 服务器上创建目录。

4.6.2. 函数原型

```
ftp.mkd(pathname)
```

4.6.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
pathname	是	str	-	待创建的目录文件路径

4.6.4. 返回值描述

返回创建成功的路径信息，字符串格式。

4.6.5. 示例

```
result = ftp.mkd('test-ftp')
print(result)
```

4.7. ftp.cwd

4.7.1. 接口概述

设置服务器端的当前目录。

4.7.2. 函数原型

```
ftp.cwd(pathname)
```

4.7.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
pathname	是	str	-	设置的目录路径

4.7.4. 返回值描述

返回以 250 开头的字符串，表示设置成功（如'250 Directory successfully changed.'），否则表示设置失败。

4.7.5. 示例

```
result = ftp.cwd('/test')
print(result)   # 250 Directory successfully changed.
```

4.8. ftp.quit

4.8.1. 接口概述

关闭 FTP 服务器连接。

4.8.2. 函数原型

```
ftp.quit()
```

4.8.3. 参数描述

无。

4.8.4. 返回值描述

返回以 221 开头的字符串，表示关闭连接成功（如'221 Goodbye.'），否则表示关闭连接失败。

4.8.5. 示例

```
result = ftp.quit()
print(result)   # 221 Goodbye.
```

5 文件下载

5.1. ftp.retrbinary

5.1.1. 接口概述

使用 RETR filename FTP 命令，以二进制模式将文件从 FTP 服务器下载到本地，处理二进制文件（如图片、视频、音频等）时建议使用。

5.1.2. 函数原型

ftp.retrbinary(cmd, callback, blocksize)

5.1.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
cmd	是	str	-	传输命令，由 RETR +文件名组成（中间有空格）
callback	是	function	-	接收远端文件数据的回调函数
blocksize	否	int	2048	设置每次传输的最大字节数

5.1.4. 返回值描述

返回以 226 开头的字符串，表示下载文件成功（如'226 Transfer complete'），否则表示下载文件失败。

5.1.5. 示例

```
# 打开并创建一个文件
save_fp = open('local.bin', 'wb+')
remote_file = 'remote.bin'
# 建立下载连接
res = ftp.retrbinary(f'RETR {remote_file}', save_fp.write)

# 完成下载
msg = 'Down %s to device %s.'
if res.startswith('226 Transfer complete'):
    print(msg % (filename, 'success'))
else:
    print(msg % (filename, 'failed'))
save_fp.close()
```

5.2. ftp. retrlines

5.2.1. 接口概述

使用 RETR filename FTP 命令，以文本模式下载文件，通常用于下载文本文件。

5.2.2. 函数原型

ftp.retrlines(cmd, callback)

5.2.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
cmd	是	string	-	传输命令，由 RETR +文件名组成（中间有空格）

callback	是	function	-	接收远端文件数据的回调函数

5.2.4. 返回值描述

返回以 226 开头的字符串，表示下载文件成功（如'226 Transfer complete'），否则表示下载文件失败。

5.2.5. 示例

```
# 打开并创建一个文件
save_fp = open('local.txt', 'w+')
remote_file = 'remote.bin'
# 建立下载连接
res = ftp.retrlines(f'RETR {remote_file}', save_fp.write)

# 完成下载
msg = 'Down %s to device %s.'
if res.startswith('226 Transfer complete'):
    print(msg % (filename, 'success'))
else:
    print(msg % (filename, 'failed'))
save_fp.close()
```

6 文件上传

6.1. ftp.storbinary

6.1.1. 接口概述

以二进制形式向 FTP 服务器上传文件。

6.1.2. 函数原型

```
ftp.storbinary(cmd, fp, blocksize)
```

6.1.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
cmd	是	str	-	传输命令，由 STOR +文件名组成（中间有空格）
fp	是	file object	-	本地文件句柄（以二进制模式打开）

blocksize	否	int	2048	每次读取文件的最大字节数

6.1.4. 返回值描述

返回以 226 开头的字符串，表示上传文件成功（如'226 Transfer complete'），否则表示上传文件失败。

6.1.5. 示例

```
filename = 'ftp_file.bin'
with open(path + filename, 'rb') as fp:
# cmd 应为 STOR 命令。fp 是一个文件对象 (以二进制模式打开)。
res = ftp.storbinary('STOR ' + filename, fp)
msg = 'Upload %s to FTP Server %s.'
if res.startswith('226 Transfer complete'):
   print(msg % (filename, 'success'))
   return True
else:
   print(msg % (filename, 'falied'))
```

6.2. ftp.storlines

6.2.1. 接口概述

以 ASCII 码形式向 FTP 服务器上传文件。

6.2.2. 函数原型

```
ftp.storbinary(cmd, fp)
```

6.2.3. 参数描述

参数名	是否必填	数据类型	默认值	说明
cmd	是	str	-	传输命令，由 STOR +文件名组成（中间有空格）
fp	是	file object	-	本地文件句柄

6.2.4. 返回值描述

返回以 226 开头的字符串，表示上传文件成功（如'226 Transfer complete'），否则表示上传文件失败。

6.2.5. 示例

```
filename = 'ftp_file.bin'
with open(path + filename, 'r') as fp
```

```
res = ftp.storbinary('STOR ' + filename, fp)
msg = 'Upload %s to FTP Server %s.'
if res.startswith('226 Transfer complete'):
    print(msg % (filename, 'success'))
    return True
else:
    print(msg % (filename, 'falied'))
```