

# FTPSEVER

---

FTPSever implements a simple PUP FTP server protocol for a Xerox workstation. The server is typically run as a background process on one machine to allow other machines remote access to the files on its disk.

## Requirements

---

Ethernet connection to a remote host.

## Installation

---

Load `FTPSEVER .LCOM` from the library.

## Functions

---

(`FTPSEVER` *FTPDEBUGLOG*)

[Function]

Creates a process named `FTPSEVER` that listens on the standard PUPFTP server socket for incoming connection requests. When one arrives, `FTPSEVER` services it, then returns to its listening state. The process continues to run until killed.

If *FTPDEBUGLOG* is non-NIL, it should be an open file/stream to which tracing information is printed during the life of the process.

If *FTPDEBUGLOG* is T, output goes to a newly created window.

*FTPDEBUGLOG* can also be a REGION, specifying where the window is to be created.

`FTPSEVER.DEFAULT.HOST`

[Variable]

Initially `DSK`. This is the default host for files requested of the server via FTP. Setting this to `FLOPPY`, for example, would serve files off the machine's floppy drive.

Note: `FTPSEVER.DEFAULT.HOST` can also be set to the name of a remote host, but this has limited utility, as it does not handle passwords correctly.

## Limitations

---

The current implementation is a simple tool which allows file transfer between Xerox machines and supports only one remote connection at a time. Because of this, files cannot be loaded indirectly, i.e., via the filecoms of another file.

For example, suppose `FOO` loads `BAR` which loads `WOO`. When `FOO` is being loaded, it will attempt to load `BAR`. But FTPSever cannot support the second connection

required to load `BAR` while the first connection is still open to load `FOO`. (This is similar to the case of trying to load `FOO` and `BAR` when they are on different floppies.)

Therefore, you should load files in an order that prevents recursive loads: in this example, load `WOO`, then `BAR`, then `FOO`.

Delete (`DELFIL`) operation is now supported. Rename (`RENAMEFILE`) operation is not implemented. FTPServer is best suited for simple `COPYFILE` operations.

## Examples

---

An alternative way of specifying the host from the remote machine is to make the host name be the "device" field of the file name specification.

For example, if machine `M` is running FTPServer, another machine could ask for directory of `{M}FLOPPY:FOO.*` to get a listing of `M`'s `{FLOPPY}FOO*`.

To address your host, you may use the results of `ETHERHOSTNAME`. If on your host (`ETHERHOSTNAME NIL T`) evaluates to `123#456#`, then on a remote machine you can access file `FOO` on the host by:

```
{123#456#}FOO
```

[This page intentionally left blank]