

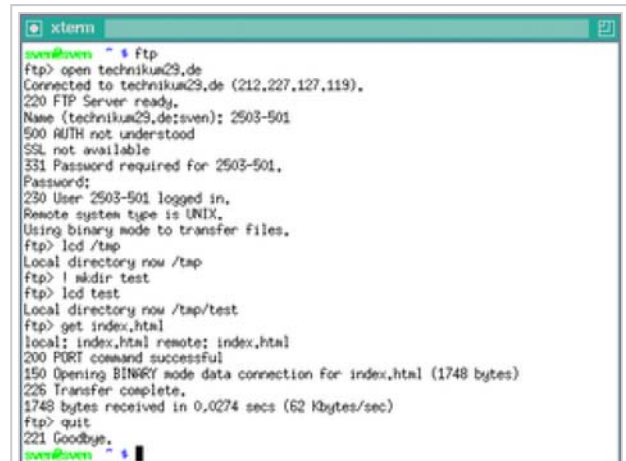
# ftp (Software)

aus Wikipedia, der freien Enzyklopädie

**ftp** ist ein fast auf jedem Betriebssystem verfügbarer interaktiver Terminal-Client für das Datenübertragungsprotokoll File Transfer Protocol (FTP). Der ursprünglich für Unix programmierte Client wurde schon bald auf andere Betriebssysteme (z. B. Windows NT) portiert.

## Inhaltsverzeichnis

- 1 Benutzung und Funktionen
- 2 Kommandos
- 3 Implementierungen
- 4 Siehe auch
- 5 Weblinks



```
xterm
even@even ~$ ftp
ftp> open technik29.de
Connected to technik29.de (212.227.127.119).
220 FTP Server ready.
Name (technik29.de:even): 2503-501
500 AUTH not understood
SSL not available
331 Password required for 2503-501.
Password:
230 User 2503-501 logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> lcd /tap
Local directory now /tap
ftp> mkdir test
ftp> lcd test
Local directory now /tap/test
ftp> get index.html
local: index.html remote: index.html
200 PORT command successful
150 Opening BINARY mode data connection for index.html (1748 bytes)
226 Transfer complete.
1748 bytes received in 0,0274 secs (62 kbytes/sec)
ftp> quit
221 Goodbye.
even@even ~$
```

Screenshot von ftp auf einem Unix-System

## Benutzung und Funktionen

Das Programm *ftp* wird meist mit einem Kommandozeilenbefehl gestartet, oft mit Angabe eines FTP-Servers als Argument, zu dem man Kontakt aufnehmen möchte:

```
ftp ftp.example.com
```

Bei erfolgreicher Verbindung fragt das Programm dann üblicherweise nach Benutzernamen und Passwort und gelangt nach abgeschlossenem Login in den Zustand, in dem es, einer Shell ähnlich, Kommandos vom Benutzer erwartet. Diese Kommandos werden jeweils mit der Enter-Taste bestätigt und anschließend ausgeführt. Erst nach der vollständigen Ausführung eines Kommandos erscheint wieder die Befehlszeile und der Benutzer kann fortfahren, Kommandos einzugeben.

Der Client ftp erlaubt das Definieren von sogenannten Makros, die dazu dienen eine komplexe Befehlskette mit einem einzigen Befehl auszuführen. Das Erstellen eines Makros geht folgendermaßen vor sich: Der Benutzer startet die Aufnahme des Makros mit dem Befehl `macdef makroname`. Daraufhin werden alle Zeilen, die der Benutzer eingibt aufgezeichnet, bis er die Aufzeichnung mit einer Leerzeile beendet. Die aufgezeichneten Befehle werden ausgeführt, sobald `$makroname` eingegeben wird. Es können höchstens 16 Makros definiert werden, die insgesamt nicht mehr als 4096 Zeichen enthalten dürfen.

## Kommandos

Die Eingabe der Befehle erfolgt analog zu der Eingabe von Befehlen in einem Terminal. Folgende

Kommandos stehen zur Verfügung:

**! [command]**

Führt ein Shell-Kommando aus bzw. wechselt zur Shell (ohne **[command]**).

**account**

Sendet das account Kommando an den Server.

**append [local\_file] [remote\_file]**

Kopiert den Inhalt einer lokalen Datei **[local\_file]** an das Ende der Datei **[remote\_file]** auf dem Server.

**ascii**

Einstellen von ASCII als Übertragungsmodus. Dies ist der Standardübertragungsmodus. Dateien, die in diesem Modus übertragen werden, erfahren eine Konvertierung zwischen den verschiedenen Zeilenumbruchvarianten der an der Übertragung beteiligten Betriebssysteme. Dieser Modus ist nur sinnvoll, wenn zwei Betriebssysteme mit unterschiedlicher Zeilenumbruchkodierung als Server bzw. Client agieren.

**binary**

Einstellen von binary als Übertragungsmodus. Der Standardübertragungsmodus ist **ASCII**. Für Binärdateien sollte zu diesem Modus umgeschaltet werden, damit nicht versehentlich zufällige Bytekombinationen, die die zu konvertierenden Zeilenumbruch-Bytes darstellen, verändert werden und damit die Binärdatei im schlimmsten Fall unbrauchbar gemacht wird.

**bye**

Schließt die Verbindung zum Server und beendet das Programm.

**cd [remote\_directory]**

Wechselt in das Arbeitsverzeichnis *[remote\_directory]* auf dem Server. Vgl. das gleichlautende Unix-Kommando `cd`

**cdup**

Wechselt in die nächsthöhere Verzeichnisebene auf dem Server. Vgl. das ähnlichlautende Unix-Kommando `cd ..`

**chmod [datei]**

Verändert die Unix-Dateirechte der Datei *[datei]*, der Syntax entspricht dabei dem des Unix-Kommandos `chmod`.

**close**

Schließt die Verbindung zum Server und löscht alle Makros. Das ftp-Programm wird jedoch nicht beendet, sodass man anschließend beispielsweise mit dem *open*-Kommando eine neue FTP-Verbindung herstellen kann.

**delete [remote\_file]**

Löscht die Datei **[remote\_file]** auf dem Server.

**dir**

Zeigt den Inhalt des aktuellen Arbeitsverzeichnisses auf dem Server an. Gleichlautend mit dem Befehl der Microsoft Windows-Eingabeaufforderung *dir*. Äquivalent zu dem Befehl *ls*

**disconnect**

Äquivalent zu **close**.

**get [remote\_file] ([local\_file])**

Kopiert die Datei **[remote\_file]** vom Server auf den Client und speichert sie unter dem Namen **[local\_file]**. Ist **[local\_file]** nicht gegeben, wird sie unter dem ursprünglichen Namen abgespeichert.

**hash**

Stellt ein, dass für jeden übertragenen Datenblock ein Doppelkreuz (#) ausgegeben werden soll. Sinnvoll bei der Übertragung von großen Dateien, da *ftp* ansonsten keinerlei Informationen über den Fortschritt der Datenübertragung ausgibt.

**help ([Kommando])**

Gibt Informationen über das Client-Kommando *[Kommando]* aus. Ist kein Kommando gegeben, wird eine Liste aller verfügbaren Kommandos ausgegeben. Siehe auch *remotehelp*.

**lcd** (**[local\_directory]**)

Wechselt in das Verzeichnis **[local\_directory]** auf dem Client. Ist kein Verzeichnis angegeben, wird in das Heimatverzeichnis des aktuellen Benutzers gewechselt. Die Funktionsweise ist auf diese Weise synonym zu dem Unix-Kommando `cd`.

**ls** **[remote\_directory]** (**[local\_file]**)

Gibt eine Kurzform des Verzeichnisses **[remote\_directory]** aus und leitet die Ausgabe, wenn gegeben, in die Datei **[local\_file]** auf dem Client-Rechner um. Vgl. das gleichlautende Unix-Kommando `ls`

**mget** **[remote\_files]**

Kopiert mehrere Dateien vom Server auf den Client. Wildcards sind erlaubt. Im interaktiven Modus muss eine Bestätigung jedes Dateitransfers erfolgen.

**mkdir** **[remote\_directory]**

Erstellt das Verzeichnis **[remote\_directory]** auf dem Server. Vgl. das gleichlautende Unix-Kommando `mkdir`

**mput** **[local\_files]**

Kopiert mehrere Dateien vom Client auf den Server. Wildcards sind erlaubt. Im interaktiven Modus muss eine Bestätigung jedes Dateitransfers erfolgen.

**open** **[host]** (**[port]**)

Baut eine Verbindung zum FTP-Server *[host]* auf TCP-Port *[port]* auf.

**passive**

Schaltet den passiven FTP-Modus ein bzw. aus.

**prompt**

Schaltet den interaktiven Modus ein/aus, der bei einigen Kommandos (z.B. `mget`) zu Nachfragen führt. Standardmäßig wird der Benutzer gefragt.

**put** **[local\_file]** (**[remote\_file]**)

Kopiert eine Datei **[local\_file]** zum Server und legt sie dort unter dem Namen **[remote\_file]** ab. Ist **[remote\_file]** nicht gegeben, wird der ursprüngliche Dateiname beibehalten.

**pwd**

Gibt das aktuelle Arbeitsverzeichnis auf dem Server aus. Vgl. das gleichlautende Unix-Kommando `pwd`.

**quit**

Äquivalent zu **bye**.

**remotehelp** (**[Kommando]**)

Gibt Informationen über das Server-Kommando **[Kommando]** aus. Ist kein Kommando gegeben, wird eine Liste aller verfügbaren Kommandos ausgegeben. Siehe auch *help*.

**rename** **[from]** **[to]**

Benennt die auf dem Server liegende Datei *[from]* in *[to]* um.

**rmdir** **[remote\_directory]**

Löscht das Verzeichnis *[remote\_directory]* auf dem Server. Wie üblich auf Unix-Systemen können nur leere Verzeichnisse gelöscht werden. Daher muss der Client vorher rekursiv das Verzeichnis durchgehen und alle Dateien darin löschen.

**runique**

Verbietet das Überschreiben von Dateien auf dem Client-Rechner, indem an den Dateinamen eine Endung in Form von *.Zahl* angehängt wird.

**send** **[local\_file]** (**[remote\_file]**)

Äquivalent zu *put*.

**status**

Gibt Statusinformationen aus.

**sunique**

Wie *runique* nur für Dateien auf dem Server.

**type** **[type]**

Festlegen des Übertragungsmodus (ASCII,IMAGE). Fehlt *[type]*, wird der aktuell verwendete Übertragungsmodus ausgegeben. Vgl. die Kommandos *binary* und *ascii* weiter oben, die die gleiche Aufgabe erfüllen.

**user** **[username]** **[password]**

Anmelden am FTP-Server mit dem Benutzernamen **[username]** und dem Passwort **[password]**.

## Implementierungen

Es existieren diverse Implementierungen für verschiedenste Unix-Derivate sowie für Microsoft Windows ab Version 95.

## Siehe auch

- File Transfer Protocol

## Weblinks

- `ftp(1)` (<http://linux.die.net/man/1/ftp>) – Linux-Manpage

Von „[http://de.wikipedia.org/w/index.php?title=Ftp\\_\(Software\)&oldid=127530556](http://de.wikipedia.org/w/index.php?title=Ftp_(Software)&oldid=127530556)“

Kategorien: FTP-Client | Freie FTP-Software

- 
- Diese Seite wurde zuletzt am 13. Februar 2014 um 16:16 Uhr geändert.
  - Abrufstatistik

Der Text ist unter der Lizenz „Creative Commons Attribution/Share Alike“ verfügbar; Informationen zum Lizenzstatus eingebundener Mediendateien (etwa Bilder oder Videos) können im Regelfall durch Anklicken dieser abgerufen werden. Möglicherweise unterliegen die Inhalte jeweils zusätzlichen Bedingungen. Durch die Nutzung dieser Website erklären Sie sich mit den Nutzungsbedingungen und der Datenschutzrichtlinie einverstanden.

Wikipedia® ist eine eingetragene Marke der Wikimedia Foundation Inc.