# FTP Implementation

Generated by Doxygen 1.8.12

# Contents

# Chapter 1

# Class Index

## 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

# Chapter 2

# File Index

## 2.1 File List

Here is a list of all documented files with brief descriptions:

# Chapter 3

# Class Documentation

## 3.1 Handler Struct Reference

### Public Attributes

- pf_check **check**
- pf_handle **handle**

The documentation for this struct was generated from the following file:

- common.h

## 3.2 Pasv_info Struct Reference

### Public Attributes

- int **a**
- int **b**
- int **c**
- int **d**
- int **p**
- int **q**

The documentation for this struct was generated from the following file:

- common.h

## 3.3 thread_data Struct Reference

### Public Attributes

- pthread_t **tid**
- int **connfd**
- char * **user**
- char * **type**
- char * **prefix**
- char * **temp**
- struct sockaddr_in * **remote**
- int **listen**
- unsigned long **status**

The documentation for this struct was generated from the following file:

- common.h

# Chapter 4

# File Documentation

## 4.1 common.c File Reference

implement common functions

```
#include "common.h"
```

### Functions

- void clean_thd (thread_data *thd)
- int attr (thread_data *thd, unsigned long flag)
- void setattr (thread_data *thd, unsigned long flag, int method)
- void clearattr (thread_data *thd, unsigned long flag)
- void copy_to (char **dest, const void *p, size_t len)
- void set_remote (thread_data *thd, int h1, int h2, int h3, int h4, int p1, int p2)
- int write_b (int fd, const char *sentence, size_t length)
- int check_path (const char *path)

### 4.1.1 Detailed Description

implement common functions

### 4.1.2 Function Documentation

#### 4.1.2.1 attr()

```
int attr (
            thread_data * thd,
            unsigned long flag )
```

Test if thd's status satisfy flag

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *flag* | one flag or the bit or of flags |

**Returns**

bool indicator

#### 4.1.2.2 check_path()

```
int check_path (
            const char * path )
```

check if a path is valid

**Parameters**

| | |
|---|---|
| *path* | the path |

**Returns**

IS_FILE or IS_INVALID or IS_DIR

#### 4.1.2.3 clean_thd()

```
void clean_thd (
            thread_data * thd )
```

As destructor of struct hread_data

**Parameters**

| | |
|---|---|
| *thd* | the object |

#### 4.1.2.4 clearattr()

```
void clearattr (
            thread_data * thd,
            unsigned long flag )
```

Clear certain attributes in thread_data

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *flag* | attributes |

#### 4.1.2.5 copy_to()

```
void copy_to (
            char ** dest,
            const void * p,
            size_t len )
```

Free resource and make a copy of p stored at ∗dest

**Parameters**

| | |
|---|---|
| *dest* | pointer's pointer |
| *p* | source data pointer |
| *len* | length in byte |

#### 4.1.2.6 set_remote()

```
void set_remote (
            thread_data * thd,
            int h1,
            int h2,
            int h3,
            int h4,
            int p1,
            int p2 )
```

For PORT command, set a sockaddr_in struct using given parameters

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *h1* | ip(1) |
| *h2* | ip(2) |
| *h3* | ip(3) |
| *h4* | ip(4) |
| *p1* | port(higher 8-bit) |
| *p2* | port(lower 8-bit) |

#### 4.1.2.7 setattr()

```
void setattr (
            thread_data * thd,
            unsigned long flag,
            int method )
```

Set thread_data's status

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *flag* | flag to set |
| *method* | 0 means bit or, 1 means assignment |

#### 4.1.2.8 write_b()

```
int write_b (
            int fd,
            const char * sentence,
            size_t length )
```

Binary write function

**Parameters**

| | |
|---|---|
| *fd* | file descriptor |
| *sentence* | data to send |
| *length* | length of data |

**Returns**

length or -1 indicating error

## 4.2 handlers.c File Reference

implement handlers

```
#include "common.h"
```

### Functions

- int **user_check** (const char ∗st)
- int **pass_check** (const char ∗st)
- int **syst_check** (const char ∗st)
- int **type_check** (const char ∗st)
- int **quit_check** (const char ∗st)
- int **abor_check** (const char ∗st)
- int **port_check** (const char ∗st)
- int **pasv_check** (const char ∗st)
- int **retr_check** (const char ∗st)
- int **stor_check** (const char ∗st)
- int **cwd_check** (const char ∗st)
- int **pwd_check** (const char ∗st)
- int **cdup_check** (const char ∗st)
- int **dele_check** (const char ∗st)
- int **mkd_check** (const char ∗st)
- int **rmd_check** (const char ∗st)
- int **rnfr_check** (const char ∗st)
- int **rnto_check** (const char ∗st)
- int **list_check** (const char ∗st)
- int user_handle (thread_data ∗thd, char ∗st)
- int pass_handle (thread_data ∗thd, char ∗st)
- int syst_handle (thread_data ∗thd, char ∗st)
- int type_handle (thread_data ∗thd, char ∗st)

- int quit_handle (thread_data *thd, char *st)
- int port_handle (thread_data *thd, char *st)
- int pasv_handle (thread_data *thd, char *st)
- void send_file (thread_data *thd, char *st, int fd, FILE *fp)
- int retr_handle (thread_data *thd, char *st)
- int list_handle (thread_data *thd, char *st)
- void get_file (thread_data *thd, char *st, int fd, FILE *fp)
- int stor_handle (thread_data *thd, char *st)
- int cwd_handle (thread_data *thd, char *st)
- int cdup_handle (thread_data *thd, char *st)
- int pwd_handle (thread_data *thd, char *st)
- int dele_handle (thread_data *thd, char *st)
- int mkd_handle (thread_data *thd, char *st)
- int rmd_handle (thread_data *thd, char *st)
- int rnfr_handle (thread_data *thd, char *st)
- int rnto_handle (thread_data *thd, char *st)
- Handler _register (pf_check c, pf_handle h)
- void Register_Handlers (Handler **arr)

## Variables

- const size_t **handler_count** = sizeof(all_check_in_turn) / sizeof(pf_check)

### 4.2.1 Detailed Description

implement handlers

### 4.2.2 Function Documentation

#### 4.2.2.1 _register()

```
Handler _register (
            pf_check c,
            pf_handle h )
```

Helper function for register a Handler

**Parameters**

| | |
|---|---|
| *c* | check function |
| *h* | handle function |

**Returns**

Handler object

#### 4.2.2.2 cdup_handle()

```
int cdup_handle (
            thread_data * thd,
            char * st )
```

CDUP command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

250 on success else 550

#### 4.2.2.3 cwd_handle()

```
int cwd_handle (
            thread_data * thd,
            char * st )
```

CWD command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

250 on success else 550

#### 4.2.2.4 dele_handle()

```
int dele_handle (
            thread_data * thd,
            char * st )
```

DELE command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

250 on success else 550

#### 4.2.2.5 get_file()

```
void get_file (
            thread_data * thd,
            char * st,
            int fd,
            FILE * fp )
```

Helper function for getting a file from client

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |
| *fd* | file descriptor |
| *fp* | file pointer |

#### 4.2.2.6 list_handle()

```
int list_handle (
            thread_data * thd,
            char * st )
```

LIST command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

info about file or directory in /bin/ls format: written to data connection

#### 4.2.2.7 mkd_handle()

```
int mkd_handle (
            thread_data * thd,
            char * st )
```

MKD command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

257 and maked dir on success

#### 4.2.2.8 pass_handle()

```
int pass_handle (
            thread_data * thd,
            char * st )
```

PASS command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

230 if logged in

#### 4.2.2.9 pasv_handle()

```
int pasv_handle (
            thread_data * thd,
            char * st )
```

PASV command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

227, ip and port

#### 4.2.2.10 port_handle()

```
int port_handle (
            thread_data * thd,
            char * st )
```

PORT command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

200 if set successfully

#### 4.2.2.11 pwd_handle()

```
int pwd_handle (
            thread_data * thd,
            char * st )
```

PWD command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

257 and current prefix

#### 4.2.2.12 quit_handle()

```
int quit_handle (
            thread_data * thd,
            char * st )
```

QUIT command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

useless

#### 4.2.2.13 Register_Handlers()

```
void Register_Handlers (
            Handler ** arr )
```

Register all handlers

**Parameters**

| | |
|---|---|
| *arr* | container for Handlers |

#### 4.2.2.14 retr_handle()

```
int retr_handle (
            thread_data * thd,
            char * st )
```

RETR command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

226 on finishing transmission

#### 4.2.2.15 rmd_handle()

```
int rmd_handle (
            thread_data * thd,
            char * st )
```

RMD command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

250 on success else 550 or 504

#### 4.2.2.16 rnfr_handle()

```
int rnfr_handle (
            thread_data * thd,
            char * st )
```

RNFR command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

350 on success else 550

#### 4.2.2.17 rnto_handle()

```
int rnto_handle (
            thread_data * thd,
            char * st )
```

RNTO command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

250 on success else 503

#### 4.2.2.18 send_file()

```
void send_file (
            thread_data * thd,
            char * st,
            int fd,
            FILE * fp )
```

Helper function for sending a file to client

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |
| *fd* | file descriptor |
| *fp* | file pointer |

#### 4.2.2.19 stor_handle()

```
int stor_handle (
            thread_data * thd,
            char * st )
```

STOR command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

226 on finishing transmission

#### 4.2.2.20 syst_handle()

```
int syst_handle (
            thread_data * thd,
            char * st )
```

SYST command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

215 everytime

#### 4.2.2.21 type_handle()

```
int type_handle (
            thread_data * thd,
            char * st )
```

TYPE command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

200 and the new type

#### 4.2.2.22 user_handle()

```
int user_handle (
            thread_data * thd,
            char * st )
```

USER command handler

**Parameters**

| | |
|---|---|
| *thd* | thread_data |
| *st* | command |

**Returns**

331 if ready to log in

## 4.3 server.c File Reference

implement server

```
#include "common.h"
```

### Functions

- void process_exit (int)
- ssize_t read_s (thread_data *, char *)
- int handle_s (thread_data *, char *)
- void * dispatcher (void *)
- int **main** (int argc, char **argv)
- Pasv_info pasv_init (thread_data *thd)
- void thread_exit (thread_data *d)
- void write_s (thread_data *thd, const char *sentence, size_t length)

### Variables

- int **listen_port** = 21
- const char * **root_path** = "/tmp"

### 4.3.1 Detailed Description

implement server

### 4.3.2 Function Documentation

#### 4.3.2.1 dispatcher()

```
void * dispatcher (
            void * arg )
```

Dispatch all operation messages. Exit if no more messages or on error.

**Parameters**

| | |
|---|---|
| *arg* | thread_data of the thread |

**Returns**

always NULL

#### 4.3.2.2 handle_s()

```
int handle_s (
            thread_data * thd,
            char * sentence )
```

Handle operation string

**Parameters**

| | |
|---|---|
| *thd* | current thread |
| *sentence* | operation |

**Returns**

response stirng's length

#### 4.3.2.3 pasv_init()

```
Pasv_info pasv_init (
            thread_data * thd )
```

Get host ip, choose a port, and return

**Parameters**

| | |
|---|---|
| *thd* | current thread |

**Returns**

(a, b, c, d, p, q) indicates a.b.c.d:(p ∗ 256 + q)

#### 4.3.2.4 process_exit()

```
void process_exit (
            int sig )
```

Join all threads and exit main process.

**Parameters**

| | |
|---|---|
| *sig* | signal Keyboard_Interrupt |

#### 4.3.2.5 read_s()

```
ssize_t read_s (
            thread_data * thd,
            char * dest )
```

Read a string from a socket in a thread.

**Parameters**

| | |
|---|---|
| *thd* | current thread |
| *dest* | destination to store the string |

**Returns**

received string length (exclude '\0')

#### 4.3.2.6 thread_exit()

```
void thread_exit (
            thread_data * d )
```

Exit a thread and free resource.

**Parameters**

| | |
|---|---|
| *d* | thread descriptor to exit |

#### 4.3.2.7 write_s()

```
void write_s (
            thread_data * thd,
            const char * sentence,
            size_t length )
```

Write a string to a socket in a thread

**Parameters**

| | |
|---|---|
| *thd* | current thread |
| *sentence* | string to send |
| *length* | string length (exclude '\0') |

# Index