TCP client/server

Generated by Doxygen 1.9.4

Chapter 1

File Index

1.1 File List

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

Chapter 2

File Documentation

2.1 tcp.h File Reference

The header file include all functions used in program and some enums.

```
#include <stdio.h>
```

Macros

- #define MAX_BUFFER_SIZE 1024
- #define MAX_UNSIGNED_SHORT_VAL 65535
- #define DEFAULT_PROTOCOL 0
- #define SEND_FLAGS 0
- #define RECV_FLAGS 0
- #define DEFAULT_PORT 80
- #define DEFAULT_IP INADDR_LOOPBACK

Enumerations

- enum CLIENT_ERRORS {
  INCORRECT_ARG_NUM = -5 , SERVER_ERROR = -4 , INCORRECT_PORT = -3 , INCORRECT_IP = -2 ,
  CLIENT_CONNECT_ERROR = -1 , INCORRECT_SOCKET = -1 }
- enum CLI_ARGS_COUNT {
  USR_CLI_DEFAULT_IP_N_PORT = 2 , USR_CLI_DEFAULT_PORT = 3 , USR_ALL_ARGS_PASSED = 4 ,
  SERVER_CLI_DEFAULT_PORT = 2 ,
  SERVER_ALL_ARGS_PASSED = 3 }

Functions

- int client_connect (const int ip_addr, const int port)

  function that use TCP protocol to connect to network with passed ip and port
- int client_cli (const int argc, char ∗argv[ ])

  function that handle command line arguments and connect to network by using client_connect() if arguments is correct.
- int server_start (int port)

  start listen on chosen port through TCP protocol
- int server_cli (int argc, char ∗argv[ ])

  function that handle command line arguments and start server by using server_start() if argument is correct
- void **write_help** ()

  write help if help flag is used (–help)

2.1.1 Detailed Description

The header file include all functions used in program and some enums.

2.1.2 Macro Definition Documentation

2.1.2.1 DEFAULT_IP

```
#define DEFAULT_IP INADDR_LOOPBACK
```

default ip. INADDR_LOOPBACK is localhost ip.

2.1.2.2 DEFAULT_PORT

```
#define DEFAULT_PORT 80
```

default port value

2.1.2.3 DEFAULT_PROTOCOL

```
#define DEFAULT_PROTOCOL 0
```

used as third argument of socket() function

2.1.2.4 MAX_BUFFER_SIZE

```
#define MAX_BUFFER_SIZE 1024
```

max buffer size that server will receive by using recv() function

2.1.2.5 MAX_UNSIGNED_SHORT_VAL

```
#define MAX_UNSIGNED_SHORT_VAL 65535
```

max unsigned short type value is 2 ^ 16 - 1 = 65535

2.1.2.6 RECV_FLAGS

```
#define RECV_FLAGS 0
```

flags used in recv() function

2.1.2.7 SEND_FLAGS

```
#define SEND_FLAGS 0
```

flags used in send() function

2.1.3 Enumeration Type Documentation

2.1.3.1 CLI_ARGS_COUNT

```
enum CLI_ARGS_COUNT
```

Enumerator

USR_CLI_DEFAULT_IP_N_PORT	the value 2 is argument count passed through command line. If the program work like a client, that it require 4 arguments (first argument - the executable name, second is flag to indetify, that program working as a client (–client), third - ip address, fourth - port. But last two arguments can be omitted. The default ip address 127.0.0.1 (localhost), default port - 80.
USR_CLI_DEFAULT_PORT	3 arguments passed, the only missing one in port. Default port is 80. **See also** USR_CLI_DEFAULT_IP_N_PORT
USR_ALL_ARGS_PASSED	4 arguments passed, including ip address and port number. **See also** USR_CLI_DEFAULT_IP_N_PORT
SERVER_CLI_DEFAULT_PORT	2 arguments passed. If the program work as a server, that it require 3 arguments: first one is executable name, second is flag to indetify, that program working as a server (–server), third - port number. But last one can be ommited. The port will be set to default value - 80.
SERVER_ALL_ARGS_PASSED	3 arguments passed, including port number. **See also** SERVER_CLI_DEFAULT_PORT

2.1.3.2 CLIENT_ERRORS

```
enum CLIENT_ERRORS
```

Enumerator

INCORRECT_ARG_NUM	return value of client_cli() or server_cli() if incorrect number of arguments was passed
SERVER_ERROR	return value of server_start() if error was detected
INCORRECT_PORT	return value of client_cli() if incorrect port was passed
INCORRECT_IP	return value of client_cli() if incorrect ip was passed
CLIENT_CONNECT_ERROR	code of client error that will returned in client_connect() function
INCORRECT_SOCKET	if socket() function get error it return -1

2.1.4 Function Documentation

2.1.4.1 client_cli()

```
int client_cli (
            const int argc,
            char * argv[ ] )
```

function that handle command line arguments and connect to network by using client_connect() if arguments is correct.

Parameters

argc	argument count
argv	array of strings

Returns

return INCORRECT_PORT if port number is incorrect, INCORRRECT_IP if ip address is incorrect, INCORRECT_ARG_NUM if argument number is incorrect, CLIENT_CONNECT_ERROR if client_connect() return error, else return 0

2.1.4.2 client_connect()

```
int client_connect (
            const int ip_addr,
            const int port )
```

function that use TCP protocol to connect to network with passed ip and port

Parameters

ip_addr	ip address to connect
port	port number

Returns

return CLIENT_CONNECT_ERROR if socket() or sendto() return error value

2.1.4.3 server_cli()

```
int server_cli (
            int argc,
            char * argv[ ] )
```

function that handle command line arguments and start server by using server_start() if argument is correct

Parameters

argc	argument count
argv	array of strings

Returns

return INCORRECT_PORT if port number is incorrect, INCORRECT_ARG_NUM if argument number is incorrect, return SERVER_ERROR if server_start() return error, else return 0

2.1.4.4 server_start()

```
int server_start (
              int port )
```

start listen on chosen port through TCP protocol

Parameters

port	chosen port

Returns

SERVER_ERROR if a server socket creating error or bind function error has occured

2.2 tcp.h

Go to the documentation of this file.

```

#if !defined TCP_H_INCLUDED
#define TCP_H_INCLUDED

#include <stdio.h>


#define MAX_BUFFER_SIZE        1024
#define MAX_UNSIGNED_SHORT_VAL 65535
#define DEFAULT_PROTOCOL       0
#define SEND_FLAGS             0
#define RECV_FLAGS             0
#define DEFAULT_PORT 80
#define DEFAULT_IP   INADDR_LOOPBACK
enum CLIENT_ERRORS {
        INCORRECT_ARG_NUM    = -5,
        SERVER_ERROR         = -4,
        INCORRECT_PORT       = -3,
        INCORRECT_IP         = -2,
        CLIENT_CONNECT_ERROR = -1,
        INCORRECT_SOCKET     = -1,
};

enum CLI_ARGS_COUNT {
        USR_CLI_DEFAULT_IP_N_PORT = 2,
        USR_CLI_DEFAULT_PORT      = 3,
        USR_ALL_ARGS_PASSED       = 4,
        SERVER_CLI_DEFAULT_PORT   = 2,
        SERVER_ALL_ARGS_PASSED    = 3
};
```

```



int client_connect(const int ip_addr, const int port);

int client_cli(const int argc, char* argv[]);

int server_start(int port);

int server_cli(int argc, char* argv[]);


void write_help();

#endif
```

Index