

ZipZop

Generated by Doxygen 1.8.13

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	client Struct Reference	5
3.1.1	Field Documentation	5
3.1.1.1	name	5
3.1.1.2	sockfd	5
3.1.1.3	thread	5
3.2	message Struct Reference	6
3.2.1	Detailed Description	6
3.2.2	Field Documentation	6
3.2.2.1	content	6
3.2.2.2	sender_name	6
3.3	sllist Struct Reference	6
3.3.1	Field Documentation	7
3.3.1.1	key	7
3.3.1.2	next	7

4 File Documentation	9
4.1 src/client.c File Reference	9
4.1.1 Function Documentation	10
4.1.1.1 client_create()	10
4.1.1.2 client_destroy()	10
4.1.1.3 client_get_name()	10
4.1.1.4 client_get_socket()	10
4.1.1.5 client_get_thread()	10
4.1.1.6 client_set_name()	10
4.1.1.7 client_set_socket()	11
4.1.1.8 client_set_thread()	11
4.2 src/client.h File Reference	11
4.2.1 Function Documentation	12
4.2.1.1 client_create()	12
4.2.1.2 client_destroy()	12
4.2.1.3 client_get_name()	12
4.2.1.4 client_get_socket()	12
4.2.1.5 client_get_thread()	12
4.2.1.6 client_set_name()	13
4.2.1.7 client_set_socket()	13
4.2.1.8 client_set_thread()	13
4.3 src/errcodes.h File Reference	13
4.3.1 Enumeration Type Documentation	13
4.3.1.1 errcodes	13
4.4 src/message.c File Reference	14
4.4.1 Function Documentation	15
4.4.1.1 message_create()	15
4.4.1.2 message_destroy()	15
4.4.1.3 message_get_content()	16
4.4.1.4 message_get_sender()	16

4.4.1.5	message_pack()	16
4.4.1.6	message_unpack()	17
4.5	src/message.h File Reference	17
4.5.1	Function Documentation	18
4.5.1.1	message_create()	19
4.5.1.2	message_destroy()	19
4.5.1.3	message_get_content()	19
4.5.1.4	message_get_sender()	20
4.5.1.5	message_pack()	20
4.5.1.6	message_unpack()	21
4.6	src/sllist.c File Reference	21
4.6.1	Function Documentation	22
4.6.1.1	sll_get_key()	22
4.6.1.2	sll_get_next()	23
4.6.1.3	sll_init()	23
4.6.1.4	sll_insert_first()	23
4.6.1.5	sll_insert_last()	23
4.6.1.6	sll_remove_elm()	23
4.6.1.7	sll_remove_first()	23
4.6.1.8	sll_remove_last()	23
4.7	src/sllist.h File Reference	24
4.7.1	Macro Definition Documentation	25
4.7.1.1	SLL_INIT	25
4.7.2	Function Documentation	25
4.7.2.1	sll_get_key()	25
4.7.2.2	sll_get_next()	25
4.7.2.3	sll_init()	25
4.7.2.4	sll_insert_first()	25
4.7.2.5	sll_insert_last()	25
4.7.2.6	sll_remove_elm()	26

4.7.2.7	<code>sll_remove_first()</code>	26
4.7.2.8	<code>sll_remove_last()</code>	26
4.8	<code>src/zip-zop-client.c</code> File Reference	26
4.8.1	Macro Definition Documentation	27
4.8.1.1	<code>MESSAGE_LEN</code>	27
4.8.1.2	<code>PORT</code>	27
4.8.2	Function Documentation	27
4.8.2.1	<code>check_args()</code>	27
4.8.2.2	<code>communicate()</code>	27
4.8.2.3	<code>create_and_connect()</code>	28
4.8.2.4	<code>get_server_addr()</code>	28
4.8.2.5	<code>listen_thread()</code>	28
4.8.2.6	<code>main()</code>	28
4.8.2.7	<code>print_usage()</code>	28
4.8.2.8	<code>server_introduction()</code>	28
4.8.2.9	<code>show_message()</code>	28
4.8.2.10	<code>speak_thread()</code>	29
4.9	<code>src/zip-zop-server.c</code> File Reference	29
4.9.1	Macro Definition Documentation	30
4.9.1.1	<code>BACKLOG</code>	30
4.9.1.2	<code>CLIENT_NAME_LEN</code>	30
4.9.1.3	<code>MESSAGE_LEN</code>	30
4.9.1.4	<code>PORT</code>	30
4.9.2	Function Documentation	30
4.9.2.1	<code>accept_clients()</code>	30
4.9.2.2	<code>client_thread_broadcast()</code>	31
4.9.2.3	<code>client_thread_listen()</code>	31
4.9.2.4	<code>create_and_bind()</code>	31
4.9.2.5	<code>create_new_client()</code>	31
4.9.2.6	<code>get_internet_addr()</code>	31
4.9.2.7	<code>kill_client()</code>	31
4.9.2.8	<code>main()</code>	31
4.9.3	Variable Documentation	32
4.9.3.1	<code>CLIENT_LIST</code>	32
4.9.3.2	<code>CLIENT_LIST_MUTEX</code>	32

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

client	5
message	6
	Struct representing a messege sent by some sender	6
sllist	6

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

src/ client.c	9
src/ client.h	11
src/ errcodes.h	13
src/ message.c	14
src/ message.h	17
src/ slist.c	21
src/ slist.h	24
src/ zip-zop-client.c	26
src/ zip-zop-server.c	29

Chapter 3

Data Structure Documentation

3.1 client Struct Reference

Data Fields

- const char * [name](#)
- int [sockfd](#)
- pthread_t [thread](#)

3.1.1 Field Documentation

3.1.1.1 name

```
const char* client::name
```

3.1.1.2 sockfd

```
int client::sockfd
```

3.1.1.3 thread

```
pthread_t client::thread
```

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

- [src/client.c](#)

3.2 message Struct Reference

Struct representing a messege sent by some sender.

Data Fields

- const char * [content](#)
- const char * [sender_name](#)

3.2.1 Detailed Description

Struct representing a messege sent by some sender.

3.2.2 Field Documentation

3.2.2.1 content

```
const char* message::content
```

The content of the message

3.2.2.2 sender_name

```
const char* message::sender_name
```

The username of the sender

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

- src/[message.c](#)

3.3 sllist Struct Reference

Collaboration diagram for sllist:



Data Fields

- void * [key](#)
- struct [slist](#) * [next](#)

3.3.1 Field Documentation

3.3.1.1 [key](#)

```
void* slist::key
```

3.3.1.2 [next](#)

```
struct slist* slist::next
```

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

- [src/slist.c](#)

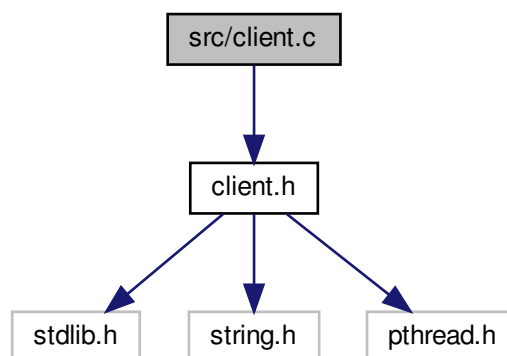
Chapter 4

File Documentation

4.1 src/client.c File Reference

```
#include "client.h"
```

Include dependency graph for client.c:



Data Structures

- struct `client`

Functions

- struct `client` * `client_create` (const char *name, int sockfd)
- void `client_destroy` (struct `client` *c)
- const char * `client_get_name` (struct `client` *c)
- int `client_get_socket` (struct `client` *c)
- pthread_t * `client_get_thread` (struct `client` *c)
- void `client_set_name` (struct `client` *c, const char *name)
- void `client_set_socket` (struct `client` *c, int sockfd)
- void `client_set_thread` (struct `client` *c, pthread_t thread)

4.1.1 Function Documentation

4.1.1.1 client_create()

```
struct client* client_create (
    const char * name,
    int sockfd )
```

4.1.1.2 client_destroy()

```
void client_destroy (
    struct client * c )
```

4.1.1.3 client_get_name()

```
const char* client_get_name (
    struct client * c )
```

4.1.1.4 client_get_socket()

```
int client_get_socket (
    struct client * c )
```

4.1.1.5 client_get_thread()

```
pthread_t* client_get_thread (
    struct client * c )
```

4.1.1.6 client_set_name()

```
void client_set_name (
    struct client * c,
    const char * name )
```


4.1.1.7 client_set_socket()

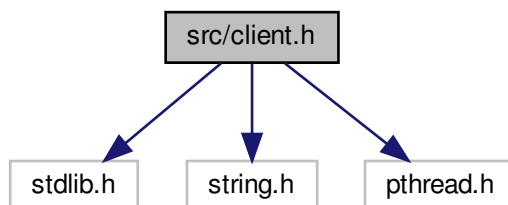
```
void client_set_socket (
    struct client * c,
    int sockfd )
```

4.1.1.8 client_set_thread()

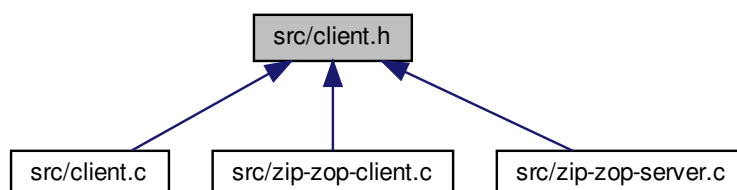
```
void client_set_thread (
    struct client * c,
    pthread_t thread )
```

4.2 src/client.h File Reference

```
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
Include dependency graph for client.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- struct `client` * `client_create` (const char *name, int sockfd)
- void `client_destroy` (struct `client` *c)
- const char * `client_get_name` (struct `client` *c)
- int `client_get_socket` (struct `client` *c)
- pthread_t * `client_get_thread` (struct `client` *c)
- void `client_set_name` (struct `client` *c, const char *name)
- void `client_set_socket` (struct `client` *c, int sockfd)
- void `client_set_thread` (struct `client` *c, pthread_t thread)

4.2.1 Function Documentation

4.2.1.1 `client_create()`

```
struct client* client_create (  
    const char * name,  
    int sockfd )
```

4.2.1.2 `client_destroy()`

```
void client_destroy (  
    struct client * c )
```

4.2.1.3 `client_get_name()`

```
const char* client_get_name (  
    struct client * c )
```

4.2.1.4 `client_get_socket()`

```
int client_get_socket (  
    struct client * c )
```

4.2.1.5 `client_get_thread()`

```
pthread_t* client_get_thread (  
    struct client * c )
```

4.2.1.6 client_set_name()

```
void client_set_name (
    struct client * c,
    const char * name )
```

4.2.1.7 client_set_socket()

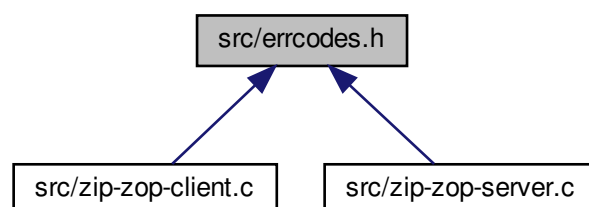
```
void client_set_socket (
    struct client * c,
    int sockfd )
```

4.2.1.8 client_set_thread()

```
void client_set_thread (
    struct client * c,
    pthread_t thread )
```

4.3 src/errcodes.h File Reference

This graph shows which files directly or indirectly include this file:



Enumerations

- enum `errcodes` {
 E_SUCCESS, E_GETADDRINFO, E_BIND, E_LISTEN,
 E_BAD_ARGS, E_CONNECT, E_PTHREAD_CREATE }

4.3.1 Enumeration Type Documentation

4.3.1.1 errcodes

```
enum errcodes
```

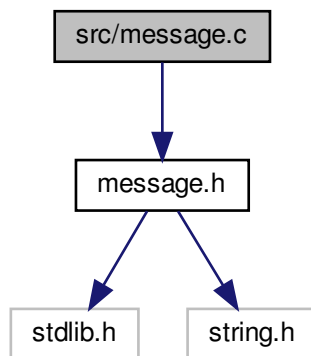
Enumerator

E_SUCCESS	
E_GETADDRINFO	
E_BIND	
E_LISTEN	
E_BAD_ARGS	
E_CONNECT	
E_PTHREAD_CREATE	

4.4 src/message.c File Reference

```
#include "message.h"
```

Include dependency graph for message.c:



Data Structures

- struct [message](#)
Struct representing a message sent by some sender.

Functions

- struct [message](#) * [message_create](#) (const char *content, const char *sender_name)
Creates a message.
- void [message_destroy](#) (struct [message](#) *m)
Destroys a message.
- const char * [message_get_content](#) (struct [message](#) *m)
Get the message content.
- const char * [message_get_sender](#) (struct [message](#) *m)
Get the message sender.
- char * [message_pack](#) (struct [message](#) *m, int *len)
Serialize a message.
- struct [message](#) * [message_unpack](#) (char *pack)

4.4.1 Function Documentation

4.4.1.1 `message_create()`

```
struct message* message_create (
    const char * content,
    const char * sender_name )
```

Creates a message.

Both parameters will be copied into the message, so the user is free to `free()` the parameters passed to this function if necessary.

Parameters

in	<i>content</i>	The content of the message.
in	<i>sender_name</i>	The username of the sender.

Returns

A pointer to a struct message in case of success, NULL otherwise. The message must be freed, using [message_destroy\(\)](#), when is not needed anymore.

See also

[message_destroy](#)

4.4.1.2 `message_destroy()`

```
void message_destroy (
    struct message * m )
```

Destroys a message.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

See also

[message_create](#)

4.4.1.3 message_get_content()

```
const char* message_get_content (
    struct message * m )
```

Get the message content.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

Returns

A pointer to the message content.

Warning

The returned value should not be freed.

4.4.1.4 message_get_sender()

```
const char* message_get_sender (
    struct message * m )
```

Get the message sender.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

Returns

A pointer to the sender name.

Warning

The returned value should not be freed.

4.4.1.5 message_pack()

```
char* message_pack (
    struct message * m,
    int * len )
```

Serialize a message.

Pack/Serialize the struct message in a format that can be sent through the network.

Parameters

in	<i>m</i>	A pointer to the message.
out	<i>len</i>	A pointer to a integer where the length of the serialized message will be stored.

Returns

A pointer to the serialized message. This should be freed when is not necessary anymore.

See also

[message_unpack](#)

4.4.1.6 message_unpack()

```
struct message* message_unpack (  
    char * pack )
```

Deserialize a message.

Unpack/Deserialize a string into a struct message.

Parameters

in	<i>pack</i>	The string that represent the packed message generated by message_pack() .
----	-------------	--

Returns

A pointer to the deserialized message. This should be freed when is not necessary anymore.

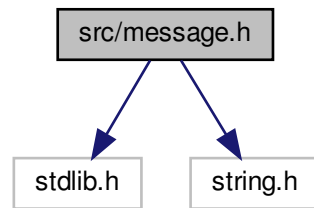
See also

[message_pack](#)

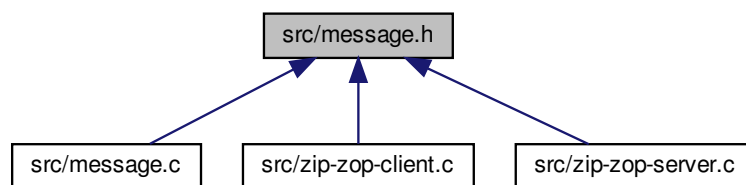
4.5 src/message.h File Reference

```
#include <stdlib.h>  
#include <string.h>
```

Include dependency graph for message.h:



This graph shows which files directly or indirectly include this file:



Functions

- struct `message` * `message_create` (const char *content, const char *sender_name)
Creates a message.
- void `message_destroy` (struct `message` *m)
Destroys a message.
- const char * `message_get_content` (struct `message` *m)
Get the message content.
- const char * `message_get_sender` (struct `message` *m)
Get the message sender.
- char * `message_pack` (struct `message` *m, int *len)
Serialize a message.
- struct `message` * `message_unpack` (char *pack)

4.5.1 Function Documentation

4.5.1.1 message_create()

```
struct message* message_create (
    const char * content,
    const char * sender_name )
```

Creates a message.

Both parameters will be copied into the message, so the user is free to `free()` the parameters passed to this function if necessary.

Parameters

in	<i>content</i>	The content of the message.
in	<i>sender_name</i>	The username of the sender.

Returns

A pointer to a struct message in case of success, NULL otherwise. The message must be freed, using [message_destroy\(\)](#), when is not needed anymore.

See also

[message_destroy](#)

4.5.1.2 message_destroy()

```
void message_destroy (
    struct message * m )
```

Destroys a message.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

See also

[message_create](#)

4.5.1.3 message_get_content()

```
const char* message_get_content (
    struct message * m )
```

Get the message content.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

Returns

A pointer to the message content.

Warning

The returned value should not be freed.

4.5.1.4 message_get_sender()

```
const char* message_get_sender (  
    struct message * m )
```

Get the message sender.

Parameters

in	<i>m</i>	A pointer to the message.
----	----------	---------------------------

Returns

A pointer to the sender name.

Warning

The returned value should not be freed.

4.5.1.5 message_pack()

```
char* message_pack (  
    struct message * m,  
    int * len )
```

Serialize a message.

Pack/Serialize the struct message in a format that can be sent through the network.

Parameters

in	<i>m</i>	A pointer to the message.
out	<i>len</i>	A pointer to a integer where the length of the serialized message will be stored.

Returns

A pointer to the serialized message. This should be freed when is not necessary anymore.

See also

[message_unpack](#)

4.5.1.6 message_unpack()

```
struct message* message_unpack (
    char * pack )
```

Deserialize a message.

Unpack/Deserialize a string into a struct message.

Parameters

in	<i>pack</i>	The string that represent the packed message generated by message_pack() .
----	-------------	--

Returns

A pointer to the deserialized message. This should be freed when is not necessary anymore.

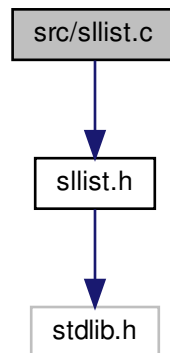
See also

[message_pack](#)

4.6 src/sllist.c File Reference

```
#include "sllist.h"
```

Include dependency graph for sllist.c:



Data Structures

- struct [sllist](#)

Functions

- struct [sllist](#) * [sll_init](#) (void)
- struct [sllist](#) * [sll_get_next](#) (struct [sllist](#) **l)
- void [sll_insert_first](#) (struct [sllist](#) **l, void *a)
- void [sll_insert_last](#) (struct [sllist](#) **l, void *a)
- void * [sll_remove_first](#) (struct [sllist](#) **l)
- void * [sll_remove_last](#) (struct [sllist](#) **l)
- void * [sll_remove_elm](#) (struct [sllist](#) **l, void *elm)
- void * [sll_get_key](#) (struct [sllist](#) *l)

4.6.1 Function Documentation

4.6.1.1 sll_get_key()

```
void* sll_get_key (  
    struct sllist * l )
```

4.6.1.2 sll_get_next()

```
struct sllist* sll_get_next (
    struct sllist ** l )
```

4.6.1.3 sll_init()

```
struct sllist* sll_init (
    void )
```

4.6.1.4 sll_insert_first()

```
void sll_insert_first (
    struct sllist ** l,
    void * a )
```

4.6.1.5 sll_insert_last()

```
void sll_insert_last (
    struct sllist ** l,
    void * a )
```

4.6.1.6 sll_remove_elm()

```
void* sll_remove_elm (
    struct sllist ** l,
    void * elm )
```

4.6.1.7 sll_remove_first()

```
void* sll_remove_first (
    struct sllist ** l )
```

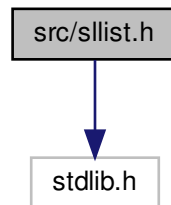
4.6.1.8 sll_remove_last()

```
void* sll_remove_last (
    struct sllist ** l )
```

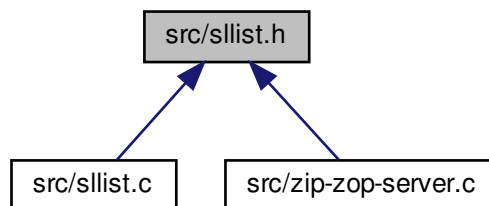
4.7 src/slist.h File Reference

```
#include <stdlib.h>
```

Include dependency graph for slist.h:



This graph shows which files directly or indirectly include this file:



Macros

- `#define SLL_INIT() NULL;`

Functions

- `struct slist * sll_init (void)`
- `struct slist * sll_get_next (struct slist **l)`
- `void sll_insert_first (struct slist **l, void *a)`
- `void sll_insert_last (struct slist **l, void *a)`
- `void * sll_remove_first (struct slist **l)`
- `void * sll_remove_last (struct slist **l)`
- `void * sll_remove_elm (struct slist **l, void *elm)`
- `void * sll_get_key (struct slist *l)`

4.7.1 Macro Definition Documentation

4.7.1.1 SLL_INIT

```
#define SLL_INIT( ) NULL;
```

4.7.2 Function Documentation

4.7.2.1 sll_get_key()

```
void* sll_get_key (
    struct sllist * l )
```

4.7.2.2 sll_get_next()

```
struct sllist* sll_get_next (
    struct sllist ** l )
```

4.7.2.3 sll_init()

```
struct sllist* sll_init (
    void )
```

4.7.2.4 sll_insert_first()

```
void sll_insert_first (
    struct sllist ** l,
    void * a )
```

4.7.2.5 sll_insert_last()

```
void sll_insert_last (
    struct sllist ** l,
    void * a )
```

4.7.2.6 sll_remove_elm()

```
void* sll_remove_elm (
    struct sllist ** l,
    void * elm )
```

4.7.2.7 sll_remove_first()

```
void* sll_remove_first (
    struct sllist ** l )
```

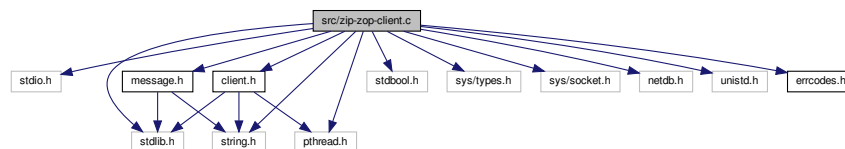
4.7.2.8 sll_remove_last()

```
void* sll_remove_last (
    struct sllist ** l )
```

4.8 src/zip-zop-client.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <unistd.h>
#include <pthread.h>
#include "errcodes.h"
#include "message.h"
#include "client.h"
```

Include dependency graph for zip-zop-client.c:



Macros

- `#define PORT "1234"`
- `#define MESSAGE_LEN 2000`

Functions

- bool [check_args](#) (int argc)
- void [print_usage](#) (const char *name)
- void [show_message](#) (struct [message](#) *m)
- void * [listen_thread](#) (void *client)
- void * [speak_thread](#) (void *client)
- struct addrinfo * [get_server_addr](#) (const char *server_name)
- int [create_and_connect](#) (struct addrinfo *addr)
- void [server_introduction](#) (struct [client](#) *c)
- void [communicate](#) (const char *user_name, int sockfd)
- int [main](#) (int argc, char **argv)

4.8.1 Macro Definition Documentation

4.8.1.1 MESSAGE_LEN

```
#define MESSAGE_LEN 2000
```

4.8.1.2 PORT

```
#define PORT "1234"
```

4.8.2 Function Documentation

4.8.2.1 check_args()

```
bool check_args (  
    int argc )
```

4.8.2.2 communicate()

```
void communicate (  
    const char * user_name,  
    int sockfd )
```

4.8.2.3 create_and_connect()

```
int create_and_connect (
    struct addrinfo * addr )
```

4.8.2.4 get_server_addr()

```
struct addrinfo* get_server_addr (
    const char * server_name )
```

4.8.2.5 listen_thread()

```
void* listen_thread (
    void * client )
```

4.8.2.6 main()

```
int main (
    int argc,
    char ** argv )
```

4.8.2.7 print_usage()

```
void print_usage (
    const char * name )
```

4.8.2.8 server_introduction()

```
void server_introduction (
    struct client * c )
```

4.8.2.9 show_message()

```
void show_message (
    struct message * m )
```

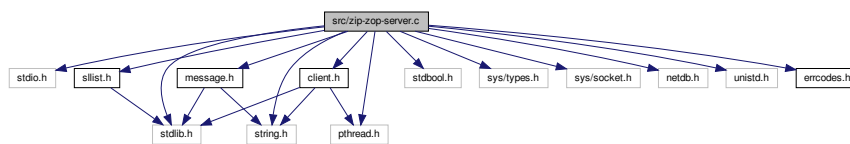
4.8.2.10 speak_thread()

```
void* speak_thread (
    void * client )
```

4.9 src/zip-zop-server.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <unistd.h>
#include <pthread.h>
#include "errcodes.h"
#include "message.h"
#include "client.h"
#include "sllist.h"
```

Include dependency graph for zip-zop-server.c:



Macros

- #define `PORT` "1234"
- #define `BACKLOG` 10
- #define `CLIENT_NAME_LEN` 100
- #define `MESSAGE_LEN` 2000

Functions

- void `client_thread_broadcast` (struct `client` *c, const char *msg)
- void `kill_client` (struct `client` *c)
- void * `client_thread_listen` (void *client)
- struct addrinfo * `get_internet_addr` (void)
- int `create_and_bind` (struct addrinfo *addr)
- void `create_new_client` (int sockfd)
- int `accept_clients` (int sockfd)
- int `main` (void)

Variables

- struct `slist` * `CLIENT_LIST` = `SLL_INIT()`
- pthread_mutex_t `CLIENT_LIST_MUTEX`

4.9.1 Macro Definition Documentation

4.9.1.1 BACKLOG

```
#define BACKLOG 10
```

4.9.1.2 CLIENT_NAME_LEN

```
#define CLIENT_NAME_LEN 100
```

4.9.1.3 MESSAGE_LEN

```
#define MESSAGE_LEN 2000
```

4.9.1.4 PORT

```
#define PORT "1234"
```

4.9.2 Function Documentation

4.9.2.1 accept_clients()

```
int accept_clients (  
    int sockfd )
```

4.9.2.2 client_thread_broadcast()

```
void client_thread_broadcast (
    struct client * c,
    const char * msg )
```

4.9.2.3 client_thread_listen()

```
void* client_thread_listen (
    void * client )
```

4.9.2.4 create_and_bind()

```
int create_and_bind (
    struct addrinfo * addr )
```

4.9.2.5 create_new_client()

```
void create_new_client (
    int sockfd )
```

4.9.2.6 get_internet_addr()

```
struct addrinfo* get_internet_addr (
    void )
```

4.9.2.7 kill_client()

```
void kill_client (
    struct client * c )
```

4.9.2.8 main()

```
int main (
    void )
```

4.9.3 Variable Documentation

4.9.3.1 CLIENT_LIST

```
struct sllist* CLIENT_LIST = SLL_INIT()
```

4.9.3.2 CLIENT_LIST_MUTEX

```
pthread_mutex_t CLIENT_LIST_MUTEX
```

Index

- accept_clients
 - zip-zop-server.c, [30](#)
- BACKLOG
 - zip-zop-server.c, [30](#)
- CLIENT_LIST_MUTEX
 - zip-zop-server.c, [32](#)
- CLIENT_LIST
 - zip-zop-server.c, [32](#)
- CLIENT_NAME_LEN
 - zip-zop-server.c, [30](#)
- check_args
 - zip-zop-client.c, [27](#)
- client, [5](#)
 - name, [5](#)
 - sockfd, [5](#)
 - thread, [5](#)
- client.c
 - client_create, [10](#)
 - client_destroy, [10](#)
 - client_get_name, [10](#)
 - client_get_socket, [10](#)
 - client_get_thread, [10](#)
 - client_set_name, [10](#)
 - client_set_socket, [10](#)
 - client_set_thread, [11](#)
- client.h
 - client_create, [12](#)
 - client_destroy, [12](#)
 - client_get_name, [12](#)
 - client_get_socket, [12](#)
 - client_get_thread, [12](#)
 - client_set_name, [12](#)
 - client_set_socket, [13](#)
 - client_set_thread, [13](#)
- client_create
 - client.c, [10](#)
 - client.h, [12](#)
- client_destroy
 - client.c, [10](#)
 - client.h, [12](#)
- client_get_name
 - client.c, [10](#)
 - client.h, [12](#)
- client_get_socket
 - client.c, [10](#)
 - client.h, [12](#)
- client_get_thread
 - client.c, [10](#)
- client.h, [12](#)
 - client.h, [12](#)
- client_set_name
 - client.c, [10](#)
 - client.h, [12](#)
- client_set_socket
 - client.c, [10](#)
 - client.h, [13](#)
- client_set_thread
 - client.c, [11](#)
 - client.h, [13](#)
- client_thread_broadcast
 - zip-zop-server.c, [30](#)
- client_thread_listen
 - zip-zop-server.c, [31](#)
- communicate
 - zip-zop-client.c, [27](#)
- content
 - message, [6](#)
- create_and_bind
 - zip-zop-server.c, [31](#)
- create_and_connect
 - zip-zop-client.c, [27](#)
- create_new_client
 - zip-zop-server.c, [31](#)
- errcodes
 - errcodes.h, [13](#)
- errcodes.h
 - errcodes, [13](#)
- get_internet_addr
 - zip-zop-server.c, [31](#)
- get_server_addr
 - zip-zop-client.c, [28](#)
- key
 - sllist, [7](#)
- kill_client
 - zip-zop-server.c, [31](#)
- listen_thread
 - zip-zop-client.c, [28](#)
- MESSAGE_LEN
 - zip-zop-client.c, [27](#)
 - zip-zop-server.c, [30](#)
- main
 - zip-zop-client.c, [28](#)
 - zip-zop-server.c, [31](#)
- message, [6](#)
 - content, [6](#)

- sender_name, 6
- message.c
 - message_create, 15
 - message_destroy, 15
 - message_get_content, 15
 - message_get_sender, 16
 - message_pack, 16
 - message_unpack, 17
- message.h
 - message_create, 18
 - message_destroy, 19
 - message_get_content, 19
 - message_get_sender, 20
 - message_pack, 20
 - message_unpack, 21
- message_create
 - message.c, 15
 - message.h, 18
- message_destroy
 - message.c, 15
 - message.h, 19
- message_get_content
 - message.c, 15
 - message.h, 19
- message_get_sender
 - message.c, 16
 - message.h, 20
- message_pack
 - message.c, 16
 - message.h, 20
- message_unpack
 - message.c, 17
 - message.h, 21
- name
 - client, 5
- next
 - sllist, 7
- PORT
 - zip-zop-client.c, 27
 - zip-zop-server.c, 30
- print_usage
 - zip-zop-client.c, 28
- SLL_INIT
 - sllist.h, 25
- sender_name
 - message, 6
- server_introduction
 - zip-zop-client.c, 28
- show_message
 - zip-zop-client.c, 28
- sll_get_key
 - sllist.c, 22
 - sllist.h, 25
- sll_get_next
 - sllist.c, 22
 - sllist.h, 25
- sll_init
 - sllist.c, 23
 - sllist.h, 25
- sll_insert_first
 - sllist.c, 23
 - sllist.h, 25
- sll_insert_last
 - sllist.c, 23
 - sllist.h, 25
- sll_remove_elm
 - sllist.c, 23
 - sllist.h, 25
- sll_remove_first
 - sllist.c, 23
 - sllist.h, 26
- sll_remove_last
 - sllist.c, 23
 - sllist.h, 26
- sllist, 6
 - key, 7
 - next, 7
- sllist.c
 - sll_get_key, 22
 - sll_get_next, 22
 - sll_init, 23
 - sll_insert_first, 23
 - sll_insert_last, 23
 - sll_remove_elm, 23
 - sll_remove_first, 23
 - sll_remove_last, 23
- sllist.h
 - SLL_INIT, 25
 - sll_get_key, 25
 - sll_get_next, 25
 - sll_init, 25
 - sll_insert_first, 25
 - sll_insert_last, 25
 - sll_remove_elm, 25
 - sll_remove_first, 26
 - sll_remove_last, 26
- sockfd
 - client, 5
- speak_thread
 - zip-zop-client.c, 28
- src/client.c, 9
- src/client.h, 11
- src/errcodes.h, 13
- src/message.c, 14
- src/message.h, 17
- src/sllist.c, 21
- src/sllist.h, 24
- src/zip-zop-client.c, 26
- src/zip-zop-server.c, 29
- thread
 - client, 5
- zip-zop-client.c
 - check_args, 27

- communicate, [27](#)
- create_and_connect, [27](#)
- get_server_addr, [28](#)
- listen_thread, [28](#)
- MESSAGE_LEN, [27](#)
- main, [28](#)
- PORT, [27](#)
- print_usage, [28](#)
- server_introduction, [28](#)
- show_message, [28](#)
- speak_thread, [28](#)
- zip-zop-server.c
 - accept_clients, [30](#)
 - BACKLOG, [30](#)
 - CLIENT_LIST_MUTEX, [32](#)
 - CLIENT_LIST, [32](#)
 - CLIENT_NAME_LEN, [30](#)
 - client_thread_broadcast, [30](#)
 - client_thread_listen, [31](#)
 - create_and_bind, [31](#)
 - create_new_client, [31](#)
 - get_internet_addr, [31](#)
 - kill_client, [31](#)
 - MESSAGE_LEN, [30](#)
 - main, [31](#)
 - PORT, [30](#)