FreeCoAP

Generated by Doxygen 1.8.9.1

Thu Sep 22 2016 18:00:53

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 coap_client_t Struct Reference

Client structure.

```
#include <coap_client.h>
```

Public Attributes

- int sd
- int timer_fd
- struct timespec timeout
- unsigned num_retrans
- struct sockaddr_in6 server_sin
- socklen_t server_sin_len
- char server_host [COAP_CLIENT_HOST_BUF_LEN]
- char server_port [COAP_CLIENT_PORT_BUF_LEN]
- coap_client_dtls_state_t state
- dtls_context_t ∗ ctx
- session_t sess
- dtls_ecdsa_key_t ecdsa_key
- char ∗ app_start
- size_t app_len

3.1.1 Detailed Description

Client structure.

3.1.2 Member Data Documentation

3.1.2.1 size_t coap_client_t::app_len

Length of application data

3.1.2.2 char∗ coap_client_t::app_start

Start of application data

3.1.2.3 dtls_context_t∗ coap_client_t::ctx

DTLS context

3.1.2.4 dtls_ecdsa_key_t coap_client_t::ecdsa_key

ECDSA keys

3.1.2.5 unsigned coap_client_t::num_retrans

Current number of retransmissions

3.1.2.6 int coap_client_t::sd

Socket descriptor

3.1.2.7 char coap_client_t::server_host[COAP_CLIENT_HOST_BUF_LEN]

String to hold the server host address

3.1.2.8 char coap_client_t::server_port[COAP_CLIENT_PORT_BUF_LEN]

String to hold the server port number

3.1.2.9 struct sockaddr_in6 coap_client_t::server_sin

IPv6 socket structture

3.1.2.10 socklen_t coap_client_t::server_sin_len

IPv6 socket structure length

3.1.2.11 session_t coap_client_t::sess

DTLS session

3.1.2.12 coap_client_dtls_state_t coap_client_t::state

Current state of the DTLS session

3.1.2.13 struct timespec coap_client_t::timeout

Timeout value

3.1.2.14 int coap_client_t::timer_fd

Timer file descriptor

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

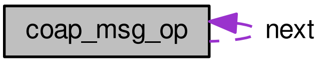
- lib/include/coap_client.h

3.2 coap_msg_op Struct Reference

Option structure.

```
#include <coap_msg.h>
```

Collaboration diagram for coap_msg_op:



Public Attributes

- unsigned num
- unsigned len
- char ∗ val
- struct coap_msg_op ∗ next

3.2.1 Detailed Description

Option structure.

3.2.2 Member Data Documentation

3.2.2.1 unsigned coap_msg_op::len

Option length

3.2.2.2 struct coap_msg_op∗ coap_msg_op::next

Pointer to the next option structure in the list

3.2.2.3 unsigned coap_msg_op::num

Option number

3.2.2.4 char∗ coap_msg_op::val

Pointer to a buffer containing the option value

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

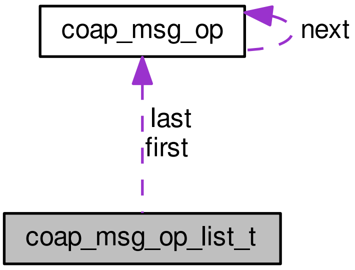
- lib/include/coap_msg.h

3.3 coap_msg_op_list_t Struct Reference

Option linked-list structure.

```
#include <coap_msg.h>
```

Collaboration diagram for coap_msg_op_list_t:



Public Attributes

- coap_msg_op_t ∗ first
- coap_msg_op_t ∗ last

3.3.1 Detailed Description

Option linked-list structure.

3.3.2 Member Data Documentation

3.3.2.1 coap_msg_op_t∗ coap_msg_op_list_t::first

Pointer to the first option structure in the list

3.3.2.2 coap_msg_op_t∗ coap_msg_op_list_t::last

Pointer to the last option structure in the list

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

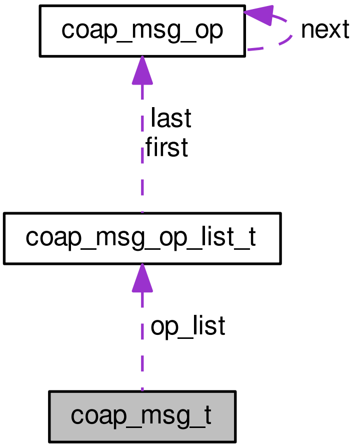
- lib/include/coap_msg.h

3.4 coap_msg_t Struct Reference

Message structure.

```
#include <coap_msg.h>
```

Collaboration diagram for coap_msg_t:



Public Attributes

- unsigned ver
- coap_msg_type_t type
- unsigned token_len
- unsigned code_class
- unsigned code_detail
- unsigned msg_id
- char token [COAP_MSG_MAX_TOKEN_LEN]
- coap_msg_op_list_t op_list
- char ∗ payload
- size_t payload_len

3.4.1 Detailed Description

Message structure.

3.4.2 Member Data Documentation

3.4.2.1 unsigned coap_msg_t::code_class

Code class

3.4.2.2 unsigned coap_msg_t::code_detail

Code detail

3.4.2.3 unsigned coap_msg_t::msg_id

Message ID

3.4.2.4 coap_msg_op_list_t coap_msg_t::op_list

Option list

3.4.2.5 char∗ coap_msg_t::payload

Pointer to a buffer containing the payload

3.4.2.6 size_t coap_msg_t::payload_len

Length of the payload

3.4.2.7 char coap_msg_t::token[COAP_MSG_MAX_TOKEN_LEN]

Token value

3.4.2.8 unsigned coap_msg_t::token_len

Token length

3.4.2.9 coap_msg_type_t coap_msg_t::type

Message type

3.4.2.10 unsigned coap_msg_t::ver

CoAP version

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

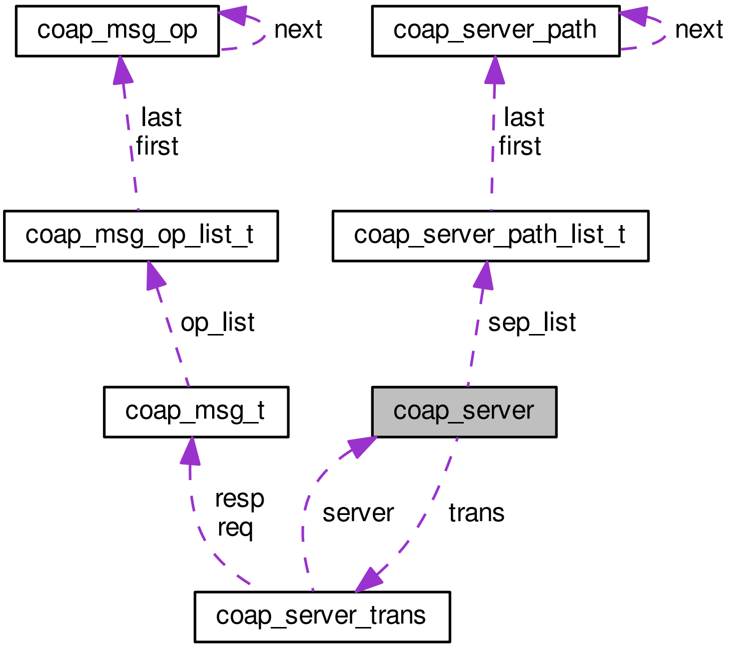
- lib/include/coap_msg.h

3.5 coap_server Struct Reference

Server structure.

```
#include <coap_server.h>
```

Collaboration diagram for coap_server:



Public Attributes

- int sd
- unsigned msg_id
- coap_server_path_list_t sep_list
- coap_server_trans_t trans [COAP_SERVER_NUM_TRANS]
- int(∗ handle)(struct coap_server ∗, coap_msg_t ∗, coap_msg_t ∗)
- dtls_ecdsa_key_t ecdsa_key

3.5.1 Detailed Description

Server structure.

3.5.2 Member Data Documentation

3.5.2.1 dtls_ecdsa_key_t coap_server::ecdsa_key

ECDSA keys

3.5.2.2 int(∗ coap_server::handle) (struct **coap_server** ∗, **coap_msg_t** ∗, **coap_msg_t** ∗)

Call-back function to handle requests and generate responses

3.5.2.3 unsigned coap_server::msg_id

Last message ID value used in a response message

3.5.2.4 int coap_server::sd

Socket descriptor

3.5.2.5 coap_server_path_list_t coap_server::sep_list

List of URI paths that require separate responses

3.5.2.6 coap_server_trans_t coap_server::trans[COAP_SERVER_NUM_TRANS]

Array of transaction structures

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

- lib/include/coap_server.h

3.6 coap_server_path Struct Reference

URI path structure.

`#include <coap_server.h>`

Collaboration diagram for coap_server_path:

coap_server_path
next

Public Attributes

- char ∗ str
- struct coap_server_path ∗ next

3.6.1 Detailed Description

URI path structure.

3.6.2 Member Data Documentation

3.6.2.1 struct coap_server_path∗ coap_server_path::next

Pointer to the next URI path structure in the list

3.6.2.2 char∗ coap_server_path::str

String containing a path

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

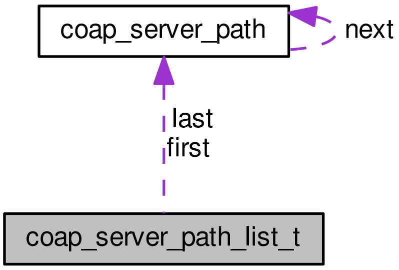
- lib/include/coap_server.h

3.7 coap_server_path_list_t Struct Reference

URI path list structure.

```
#include <coap_server.h>
```

Collaboration diagram for coap_server_path_list_t:



Public Attributes

- coap_server_path_t ∗ first
- coap_server_path_t ∗ last

3.7.1 Detailed Description

URI path list structure.

3.7.2 Member Data Documentation

3.7.2.1 coap_server_path_t∗ coap_server_path_list_t::first

Pointer to the first URI path structure in the list

3.7.2.2 coap_server_path_t∗ coap_server_path_list_t::last

Pointer to the last URI path structure in the list

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

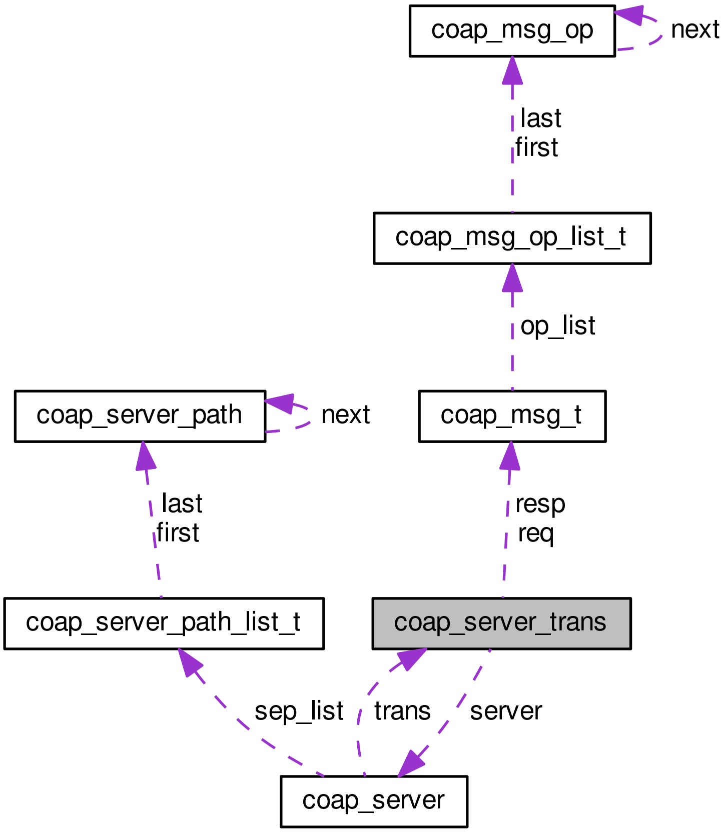
- lib/include/coap_server.h

3.8 coap_server_trans Struct Reference

Transaction structure.

```
#include <coap_server.h>
```

Collaboration diagram for coap_server_trans:



Public Attributes

- int active
- time_t last_use
- int timer_fd
- struct timespec timeout
- unsigned num_retrans
- struct sockaddr_in6 client_sin
- socklen_t client_sin_len
- char client_addr [COAP_SERVER_ADDR_BUF_LEN]
- coap_msg_t req
- coap_msg_t resp
- struct coap_server ∗ server
- coap_server_dtls_state_t state
- dtls_context_t ∗ ctx

- session_t sess
- char ∗ app_start
- size_t app_len

3.8.1 Detailed Description

Transaction structure.

3.8.2 Member Data Documentation

3.8.2.1 int coap_server_trans::active

Flag to indicate if this transaction structure contains valid data

3.8.2.2 size_t coap_server_trans::app_len

Length of application data

3.8.2.3 char∗ coap_server_trans::app_start

Start of application data

3.8.2.4 char coap_server_trans::client_addr[COAP_SERVER_ADDR_BUF_LEN]

String to hold the client address

3.8.2.5 struct sockaddr_in6 coap_server_trans::client_sin

IPv6 socket structure

3.8.2.6 socklen_t coap_server_trans::client_sin_len

IPv6 socket structure length

3.8.2.7 dtls_context_t∗ coap_server_trans::ctx

DTLS context

3.8.2.8 time_t coap_server_trans::last_use

The time that this transaction structure was last used

3.8.2.9 unsigned coap_server_trans::num_retrans

Current number of retransmissions

3.8.2.10 coap_msg_t coap_server_trans::req

Last request message received for this transaction

3.8.2.11 coap_msg_t coap_server_trans::resp

Last response message sent for this transaction

3.8.2.12 struct coap_server∗ coap_server_trans::server

Pointer to the containing server structure

3.8.2.13 session_t coap_server_trans::sess

DTLS session

3.8.2.14 coap_server_dtls_state_t coap_server_trans::state

Current state of the DTLS session

3.8.2.15 struct timespec coap_server_trans::timeout

Timeout value

3.8.2.16 int coap_server_trans::timer_fd

Timer file descriptor

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

- lib/include/coap_server.h

Chapter 4

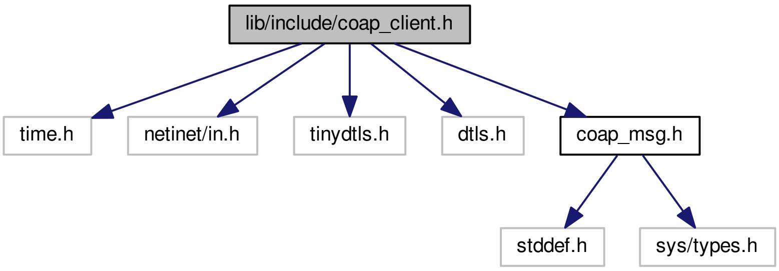
File Documentation

4.1 lib/include/coap_client.h File Reference

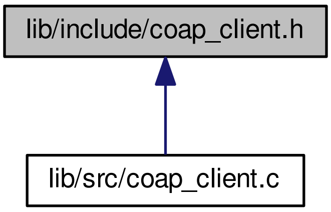
Include file for the FreeCoAP client library.

```
#include <time.h>
#include <netinet/in.h>
#include "tinydtls.h"
#include "dtls.h"
#include "coap_msg.h"
```

Include dependency graph for coap_client.h:



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



Classes

- struct coap_client_t

  Client structure.

Macros

- #define COAP_CLIENT_HOST_BUF_LEN 128
- #define COAP_CLIENT_PORT_BUF_LEN 8
- #define coap_client_dtls_get_state(client) ((client)->state)

Enumerations

- enum coap_client_dtls_state_t { COAP_CLIENT_DTLS_UNCONNECTED = 0, COAP_CLIENT_DTLS_C←ONNECTED, COAP_CLIENT_DTLS_ALERT }

  Client DTLS state enumeration.

Functions

- int coap_client_create (coap_client_t ∗client, const char ∗host, const char ∗port, const unsigned char ∗ecdsa_priv_key, const unsigned char ∗ecdsa_pub_key_x, const unsigned char ∗ecdsa_pub_key_y)

  Initialise a client structure.
- void coap_client_destroy (coap_client_t ∗client)

  Deinitialise a client structure.
- int coap_client_exchange (coap_client_t ∗client, coap_msg_t ∗req, coap_msg_t ∗resp)

  Send a request to the server and receive the response.

4.1.1 Detailed Description

Include file for the FreeCoAP client library.

4.1.2 Macro Definition Documentation

4.1.2.1 #define coap_client_dtls_get_state(*client*) ((client)->state)

Get DTLS state

4.1.2.2 #define COAP_CLIENT_HOST_BUF_LEN 128

Buffer length for host addresses

4.1.2.3 #define COAP_CLIENT_PORT_BUF_LEN 8

Buffer length for port numbers

4.1.3 Enumeration Type Documentation

4.1.3.1 enum coap_client_dtls_state_t

Client DTLS state enumeration.

Enumerator

COAP_CLIENT_DTLS_UNCONNECTED DTLS session is not active

COAP_CLIENT_DTLS_CONNECTED DTLS session is active

COAP_CLIENT_DTLS_ALERT A DTLS alert message was received from the server

4.1.4 Function Documentation

4.1.4.1 int coap_client_create (coap_client_t ∗ *client,* const char ∗ *host,* const char ∗ *port,* const unsigned char ∗ *ecdsa_priv_key,* const unsigned char ∗ *ecdsa_pub_key_x,* const unsigned char ∗ *ecdsa_pub_key_y*)

Initialise a client structure.

Parameters

out	*client*	Pointer to a client structure
in	*host*	Pointer to a string containing the host address of the server
in	*port*	Port number of the server
in	*ecdsa_priv_key*	Buffer containing the ECDSA private key
in	*ecdsa_pub_←key_x*	Buffer containing the x component of the ECDSA public key
in	*ecdsa_pub_←key_y*	Buffer containing the y component of the ECDSA public key

Returns

Operation status

Return values

0	Success
<0	Error

4.1.4.2 void coap_client_destroy (coap_client_t ∗ *client*)

Deinitialise a client structure.

Parameters

`in,out`	*client*	Pointer to a client structure

4.1.4.3 int coap_client_exchange (**coap_client_t** ∗ *client,* **coap_msg_t** ∗ *req,* **coap_msg_t** ∗ *resp*)

Send a request to the server and receive the response.

Parameters

`in,out`	*client*	Pointer to a client structure
`in`	*req*	Pointer to the request message
`out`	*resp*	Pointer to the response message

This function sets the message ID and token fields of the request message overriding any values set by the calling function.

Returns

Operation status

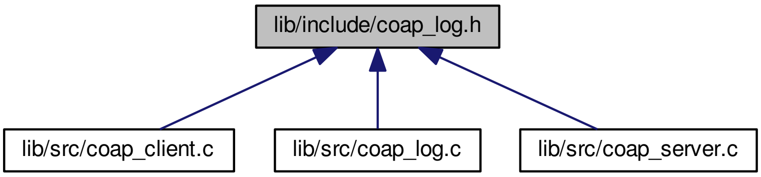
Return values

0	Success
<*0*	Error

4.2 lib/include/coap_log.h File Reference

Include file for the FreeCoAP logging module.

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



Macros

- #define COAP_LOG_DEF_LEVEL COAP_LOG_ERROR

Enumerations

- enum coap_log_level_t {
  COAP_LOG_ERROR = 0, COAP_LOG_WARN = 1, COAP_LOG_NOTICE = 2, COAP_LOG_INFO = 3,
  COAP_LOG_DEBUG = 4 }

  Log level.

Functions

- void coap_log_set_level (coap_log_level_t level)

    Set the log level.
- coap_log_level_t coap_log_get_level (void)

    Get the log level.
- void coap_log_error (const char ∗msg,...)

    Log an error message.
- void coap_log_warn (const char ∗msg,...)

    Log a warning message.
- void coap_log_notice (const char ∗msg,...)

    Log an notice message.
- void coap_log_info (const char ∗msg,...)

    Log an info message.
- void coap_log_debug (const char ∗msg,...)

    Log a debug message.

4.2.1 Detailed Description

Include file for the FreeCoAP logging module.

4.2.2 Macro Definition Documentation

4.2.2.1 #define COAP_LOG_DEF_LEVEL COAP_LOG_ERROR

Default log level

4.2.3 Enumeration Type Documentation

4.2.3.1 enum coap_log_level_t

Log level.

Enumerator

COAP_LOG_ERROR Error log level

COAP_LOG_WARN Warning log level

COAP_LOG_NOTICE Notice log level

COAP_LOG_INFO Informational log level

COAP_LOG_DEBUG Debug log level

4.2.4 Function Documentation

4.2.4.1 void coap_log_debug (const char ∗ *msg*, ...)

Log a debug message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.2.4.2 void coap_log_error (const char ∗ *msg,* *...*)

Log an error message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.2.4.3 coap_log_level_t coap_log_get_level (void)

Get the log level.

Returns

The current log level

4.2.4.4 void coap_log_info (const char ∗ *msg,* *...*)

Log an info message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.2.4.5 void coap_log_notice (const char ∗ *msg,* *...*)

Log an notice message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.2.4.6 void coap_log_set_level (coap_log_level_t *level*)

Set the log level.

Messages with a severity below this level will be filtered. Error messages cannot be filtered.

Parameters

in	*level*	The new log level

< Warning log level

< Notice warning level

< Informational warning level

< Debug warning level

4.2.4.7 void coap_log_warn (const char ∗ *msg,* *...*)

Log a warning message.

Parameters

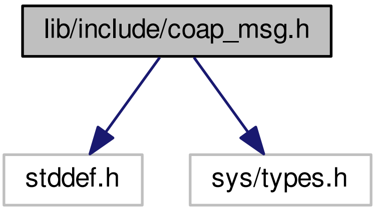
in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.3 lib/include/coap_msg.h File Reference

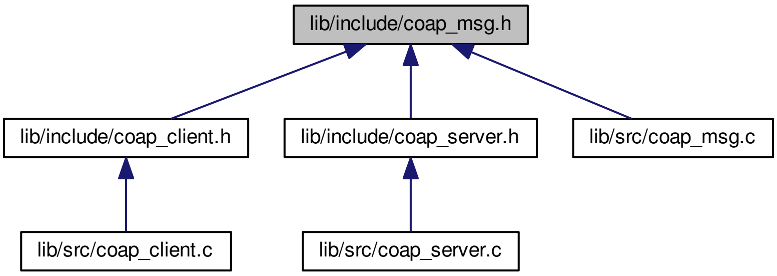
Include file for the FreeCoAP message parser/formatter library.

```
#include <stddef.h>
#include <sys/types.h>
```

Include dependency graph for coap_msg.h:



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



Classes

- struct coap_msg_op

  Option structure.
- struct coap_msg_op_list_t

  Option linked-list structure.
- struct coap_msg_t

  Message structure.

Macros

- #define COAP_MSG_VER 0x01
- #define COAP_MSG_MAX_TOKEN_LEN 8
- #define COAP_MSG_MAX_CODE_CLASS 7
- #define COAP_MSG_MAX_CODE_DETAIL 31
- #define COAP_MSG_MAX_MSG_ID ((1 << 16) - 1)
- #define COAP_MSG_OP_URI_PATH_NUM 11
- #define COAP_MSG_OP_URI_PATH_MAX_LEN 256
- #define COAP_MSG_MAX_BUF_LEN 1152
- #define coap_msg_op_num_is_critical(num) ((num) & 1)
- #define coap_msg_op_num_is_unsafe(num) ((num) & 2)
- #define coap_msg_op_num_no_cache_key(num) ((num & 0x1e) == 0x1c)
- #define coap_msg_op_get_num(op) ((op)->num)
- #define coap_msg_op_set_num(op, num) ((op)->num = (num))
- #define coap_msg_op_get_len(op) ((op)->len)
- #define coap_msg_op_set_len(op, len) ((op)->len = (len))
- #define coap_msg_op_get_val(op) ((op)->val)
- #define coap_msg_op_set_val(op, val) ((op)->val = (val))
- #define coap_msg_op_get_next(op) ((op)->next)
- #define coap_msg_op_set_next(op, next_op) ((op)->next = (next_op))
- #define coap_msg_get_ver(msg) ((msg)->ver)
- #define coap_msg_get_type(msg) ((msg)->type)
- #define coap_msg_get_token_len(msg) ((msg)->token_len)
- #define coap_msg_get_code_class(msg) ((msg)->code_class)
- #define coap_msg_get_code_detail(msg) ((msg)->code_detail)
- #define coap_msg_get_msg_id(msg) ((msg)->msg_id)
- #define coap_msg_get_token(msg) ((msg)->token)
- #define coap_msg_get_first_op(msg) ((msg)->op_list.first)
- #define coap_msg_get_payload(msg) ((msg)->payload)
- #define coap_msg_get_payload_len(msg) ((msg)->payload_len)
- #define coap_msg_is_empty(msg) (((msg)->code_class == 0) && ((msg)->code_detail == 0))

Typedefs

- typedef struct coap_msg_op coap_msg_op_t

  Option structure.

Enumerations

- enum coap_msg_type_t { COAP_MSG_CON = 0x0, COAP_MSG_NON = 0x1, COAP_MSG_ACK = 0x2, COAP_MSG_RST = 0x3 }

  Message type enumeration.
- enum coap_msg_class_t { COAP_MSG_REQ = 0, COAP_MSG_SUCCESS = 2, COAP_MSG_CLIENT_ERR = 4, COAP_MSG_SERVER_ERR = 5 }

  Code class enumeration.
- enum coap_msg_method_t { COAP_MSG_GET = 1, COAP_MSG_POST = 2, COAP_MSG_PUT = 3, COAP_MSG_DELETE =4 }

  Code detail enumeration.
- enum coap_msg_success_t {
  COAP_MSG_CREATED = 1, COAP_MSG_DELETED = 2, COAP_MSG_VALID = 3, COAP_MSG_CHANGED = 4,
  COAP_MSG_CONTENT = 5 }

Success response code detail enumeration.

- enum coap_msg_client_err_t {
COAP_MSG_BAD_REQ = 0, COAP_MSG_UNAUTHORIZED = 1, COAP_MSG_BAD_OPTION = 2, COAP_MSG_FORBIDDEN = 3,
COAP_MSG_NOT_FOUND = 4, COAP_MSG_METHOD_NOT_ALLOWED = 5, COAP_MSG_NOT_ACCEPTABLE = 6, COAP_MSG_PRECOND_FAILED = 12,
COAP_MSG_REQ_ENT_TOO_LARGE = 13, COAP_MSG_UNSUP_CONT_FMT = 15 }

  Client error response code detail enumeration.
- enum coap_msg_server_err_t {
COAP_MSG_INT_SERVER_ERR = 0, COAP_MSG_NOT_IMPL = 1, COAP_MSG_BAD_GATEWAY = 2, COAP_MSG_SERV_UNAVAIL = 3,
COAP_MSG_GATEWAY_TIMEOUT = 4, COAP_MSG_PROXY_NOT_SUP = 5 }

  Server error response code detail enumeration.
- enum coap_msg_op_num_t {
COAP_MSG_IF_MATCH = 1, COAP_MSG_URI_HOST = 3, COAP_MSG_ETAG = 4, COAP_MSG_IF_NONE_MATCH = 5,
COAP_MSG_URI_PORT = 7, COAP_MSG_LOCATION_PATH = 8, COAP_MSG_URI_PATH = 11, COAP_MSG_CONTENT_FORMAT = 12,
COAP_MSG_MAX_AGE = 14, COAP_MSG_URI_QUERY = 15, COAP_MSG_ACCEPT = 17, COAP_MSG_LOCATION_QUERY = 20,
COAP_MSG_PROXY_URI = 35, COAP_MSG_PROXY_SCHEME = 39, COAP_MSG_SIZE1 = 60 }

  Option number enumeration.

Functions

- int coap_msg_op_num_is_recognized (unsigned num)

  Check if option is recognized.
- void coap_msg_gen_rand_str (char *buf, size_t len)

  Generate a random string of bytes.
- void coap_msg_create (coap_msg_t *msg)

  Initialise a message structure.
- void coap_msg_destroy (coap_msg_t *msg)

  Deinitialise a message structure.
- void coap_msg_reset (coap_msg_t *msg)

  Deinitialise and initialise a message structure.
- unsigned coap_msg_check_critical_ops (coap_msg_t *msg)

  Check that all of the critical options in a message are recognized.
- unsigned coap_msg_check_unsafe_ops (coap_msg_t *msg)

  Check that all of the unsafe options in a message are recognized.
- int coap_msg_parse_type_msg_id (char *buf, size_t len, unsigned *type, unsigned *msg_id)

  Extract the type and message ID values from a message.
- ssize_t coap_msg_parse (coap_msg_t *msg, char *buf, size_t len)

  Parse a message.
- int coap_msg_set_type (coap_msg_t *msg, unsigned type)

  Set the type in a message.
- int coap_msg_set_code (coap_msg_t *msg, unsigned code_class, unsigned code_detail)

  Set the code in a message.
- int coap_msg_set_msg_id (coap_msg_t *msg, unsigned msg_id)

  Set the message ID in a message.
- int coap_msg_set_token (coap_msg_t *msg, char *buf, size_t len)

  Set the token in a message.
- int coap_msg_add_op (coap_msg_t *msg, unsigned num, unsigned len, const char *val)

Add a token to a message structure.

- int coap_msg_set_payload (coap_msg_t ∗msg, char ∗buf, size_t len)

    Set the payload in a message.

- ssize_t coap_msg_format (coap_msg_t ∗msg, char ∗buf, size_t len)

    Format a message.

- int coap_msg_copy (coap_msg_t ∗dst, coap_msg_t ∗src)

    Copy a message.

4.3.1 Detailed Description

Include file for the FreeCoAP message parser/formatter library.

4.3.2 Macro Definition Documentation

4.3.2.1 #define coap_msg_get_code_class(*msg*) ((msg)->code_class)

Get the code class from a message

4.3.2.2 #define coap_msg_get_code_detail(*msg*) ((msg)->code_detail)

Get the code detail from a message

4.3.2.3 #define coap_msg_get_first_op(*msg*) ((msg)->op_list.first)

Get the first option from a message

4.3.2.4 #define coap_msg_get_msg_id(*msg*) ((msg)->msg_id)

Get the message ID from message

4.3.2.5 #define coap_msg_get_payload(*msg*) ((msg)->payload)

Get the payload from a message

4.3.2.6 #define coap_msg_get_payload_len(*msg*) ((msg)->payload_len)

Get the payload length from a message

4.3.2.7 #define coap_msg_get_token(*msg*) ((msg)->token)

Get the token from a message

4.3.2.8 #define coap_msg_get_token_len(*msg*) ((msg)->token_len)

Get the token length from a message

4.3.2.9 #define coap_msg_get_type(*msg*) ((msg)->type)

Get the type from a message

4.3.2.10 #define coap_msg_get_ver(*msg*) ((msg)->ver)

Get the version from a message

4.3.2.11 #define coap_msg_is_empty(*msg*) (((msg)->code_class == 0) && ((msg)->code_detail == 0))

Indicate if a message is empty

4.3.2.12 #define COAP_MSG_MAX_BUF_LEN 1152

Maximum buffer length for header and payload

4.3.2.13 #define COAP_MSG_MAX_CODE_CLASS 7

Maximum code class

4.3.2.14 #define COAP_MSG_MAX_CODE_DETAIL 31

Maximum code detail

4.3.2.15 #define COAP_MSG_MAX_MSG_ID ((1 << 16) - 1)

Maximum message ID

4.3.2.16 #define COAP_MSG_MAX_TOKEN_LEN 8

Maximum token length

4.3.2.17 #define coap_msg_op_get_len(*op*) ((op)->len)

Get the option length from an option

4.3.2.18 #define coap_msg_op_get_next(*op*) ((op)->next)

Get the next pointer from an option

4.3.2.19 #define coap_msg_op_get_num(*op*) ((op)->num)

Get the option number from an option

4.3.2.20 #define coap_msg_op_get_val(*op*) ((op)->val)

Get the option value from an option

4.3.2.21 #define coap_msg_op_num_is_critical(*num*) ((num) & 1)

Indicate if an option is critical

4.3.2.22 #define coap_msg_op_num_is_unsafe(*num*) ((num) & 2)

Indicate if an option is unsafe to forward

4.3.2.23 #define coap_msg_op_num_no_cache_key(*num*) ((num & 0x1e) == 0x1c)

Indicate if an option is not part of the cache key

4.3.2.24 #define coap_msg_op_set_len(*op, len*) ((op)->len = (len))

Set the option length in an option

4.3.2.25 #define coap_msg_op_set_next(*op, next_op*) ((op)->next = (next_op))

Set the next pointer in an option

4.3.2.26 #define coap_msg_op_set_num(*op, num*) ((op)->num = (num))

Set the option number in an option

4.3.2.27 #define coap_msg_op_set_val(*op, val*) ((op)->val = (val))

Set the option value in an option

4.3.2.28 #define COAP_MSG_OP_URI_PATH_MAX_LEN 256

Maximum buffer length for a reconstructed URI path

4.3.2.29 #define COAP_MSG_OP_URI_PATH_NUM 11

Uri-path option number

4.3.2.30 #define COAP_MSG_VER 0x01

CoAP version

4.3.3 Enumeration Type Documentation

4.3.3.1 enum coap_msg_class_t

Code class enumeration.

Enumerator

COAP_MSG_REQ Request

COAP_MSG_SUCCESS Success response

COAP_MSG_CLIENT_ERR Client error response

COAP_MSG_SERVER_ERR Server error response

4.3.3.2 enum coap_msg_client_err_t

Client error response code detail enumeration.

Enumerator

- ***COAP_MSG_BAD_REQ*** Bad request client error
- ***COAP_MSG_UNAUTHORIZED*** Unauthorized client error
- ***COAP_MSG_BAD_OPTION*** Bad option client error
- ***COAP_MSG_FORBIDDEN*** Forbidden client error
- ***COAP_MSG_NOT_FOUND*** Not found client error
- ***COAP_MSG_METHOD_NOT_ALLOWED*** Method not allowed client error
- ***COAP_MSG_NOT_ACCEPTABLE*** Not acceptable client error
- ***COAP_MSG_PRECOND_FAILED*** Precondition failed client error
- ***COAP_MSG_REQ_ENT_TOO_LARGE*** Request entity too large client error
- ***COAP_MSG_UNSUP_CONT_FMT*** Unsupported content-format client error

4.3.3.3 enum coap_msg_method_t

Code detail enumeration.

Enumerator

- ***COAP_MSG_GET*** Get request method
- ***COAP_MSG_POST*** Post request method
- ***COAP_MSG_PUT*** Put request method
- ***COAP_MSG_DELETE*** Delete request method

4.3.3.4 enum coap_msg_op_num_t

Option number enumeration.

Enumerator

- ***COAP_MSG_IF_MATCH*** If-Match option number
- ***COAP_MSG_URI_HOST*** URI-Host option number
- ***COAP_MSG_ETAG*** Entity-Tag option number
- ***COAP_MSG_IF_NONE_MATCH*** If-None-Match option number
- ***COAP_MSG_URI_PORT*** URI-Port option number
- ***COAP_MSG_LOCATION_PATH*** Location-Path option number
- ***COAP_MSG_URI_PATH*** URI-Path option number
- ***COAP_MSG_CONTENT_FORMAT*** Content-Format option number
- ***COAP_MSG_MAX_AGE*** Max-Age option number
- ***COAP_MSG_URI_QUERY*** URI-Query option number
- ***COAP_MSG_ACCEPT*** Accept option number
- ***COAP_MSG_LOCATION_QUERY*** Location-Query option number
- ***COAP_MSG_PROXY_URI*** Proxy-URI option number
- ***COAP_MSG_PROXY_SCHEME*** Proxy-Scheme option number
- ***COAP_MSG_SIZE1*** Size1 option number

4.3.3.5 enum coap_msg_server_err_t

Server error response code detail enumeration.

Enumerator

COAP_MSG_INT_SERVER_ERR Internal server error

COAP_MSG_NOT_IMPL Not implemented server error

COAP_MSG_BAD_GATEWAY Bad gateway server error

COAP_MSG_SERV_UNAVAIL Service unavailable server error

COAP_MSG_GATEWAY_TIMEOUT Gateway timeout server error

COAP_MSG_PROXY_NOT_SUP Proxying not supported server error

4.3.3.6 enum coap_msg_success_t

Success response code detail enumeration.

Enumerator

COAP_MSG_CREATED Created success response

COAP_MSG_DELETED Deleted success response

COAP_MSG_VALID Valid success response

COAP_MSG_CHANGED Changed success response

COAP_MSG_CONTENT Content success response

4.3.3.7 enum coap_msg_type_t

Message type enumeration.

Enumerator

COAP_MSG_CON Confirmable message

COAP_MSG_NON Non-confirmable message

COAP_MSG_ACK Acknowledgement message

COAP_MSG_RST Reset message

4.3.4 Function Documentation

4.3.4.1 int coap_msg_add_op (coap_msg_t ∗ *msg*, unsigned *num*, unsigned *len*, const char ∗ *val*)

Add a token to a message structure.

Parameters

`in,out`	*msg*	Pointer to a message structure
`in`	*num*	Option number
`in`	*len*	Option length

in	*val*	Pointer to a buffer containing the option value

Returns

Operation status

Return values

0	Success
<*0*	Error

4.3.4.2 unsigned coap_msg_check_critical_ops (**coap_msg_t** ∗ *msg*)

Check that all of the critical options in a message are recognized.

Parameters

in	*msg*	Pointer to message structure

Returns

Operation status or bad option number

Return values

0	Success
>*0*	Bad option number

4.3.4.3 unsigned coap_msg_check_unsafe_ops (**coap_msg_t** ∗ *msg*)

Check that all of the unsafe options in a message are recognized.

Parameters

in	*msg*	Pointer to message structure

Returns

Operation status or bad option number

Return values

0	Success
>*0*	Bad option number

4.3.4.4 int coap_msg_copy (**coap_msg_t** ∗ *dst,* **coap_msg_t** ∗ *src*)

Copy a message.

Parameters

in,out	*dst*	Pointer to the destination message structure

in	*src*	Pointer to the source message structure

Returns

Operation status

Return values

0	Success
<*0*	Error

4.3.4.5 void coap_msg_create (coap_msg_t ∗ *msg*)

Initialise a message structure.

Parameters

out	*msg*	Pointer to a message structure

4.3.4.6 void coap_msg_destroy (coap_msg_t ∗ *msg*)

Deinitialise a message structure.

Parameters

in,out	*msg*	Pointer to a message structure

4.3.4.7 ssize_t coap_msg_format (coap_msg_t ∗ *msg*, char ∗ *buf*, size_t *len*)

Format a message.

Parameters

in	*msg*	Pointer to a message structure
out	*buf*	Pointer to a buffer to contain the formatted message
in	*len*	Length of the buffer

Returns

Length of the formatted message or error code

Return values

>*0*	Length of the formatted message
<*0*	Error

4.3.4.8 void coap_msg_gen_rand_str (char ∗ *buf*, size_t *len*)

Generate a random string of bytes.

Parameters

out	*buf*	Pointer to the buffer to store the random string
in	*len*	Length of the buffer

4.3.4.9 int coap_msg_op_num_is_recognized (unsigned *num*)

Check if option is recognized.

Parameters

in	*num*	Option number

Returns

Operation status

Return values

1	Option is recognized
0	Option is not recognized

4.3.4.10 ssize_t coap_msg_parse (coap_msg_t ∗ *msg,* char ∗ *buf,* size_t *len*)

Parse a message.

Parameters

in,out	*msg*	Pointer to a message structure
in	*buf*	Pointer to a buffer containing the message
in	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
<*0*	Error

4.3.4.11 int coap_msg_parse_type_msg_id (char ∗ *buf,* size_t *len,* unsigned ∗ *type,* unsigned ∗ *msg_id*)

Extract the type and message ID values from a message.

If a message contains a format error, this function will attempt to extract the type and message ID so that a reset message can be returned to the sender.

Parameters

in	*buf*	Pointer to a buffer containing the message
in	*len*	Length of the buffer
out	*type*	Pointer to field to store the type value
out	*msg_id*	Pointer to a field to store the message ID value

Returns

Operation status

Return values

0	Success
<0	Error

4.3.4.12 void coap_msg_reset (coap_msg_t * *msg*)

Deinitialise and initialise a message structure.

Parameters

`in,out`	*msg*	Pointer to a message structure

4.3.4.13 int coap_msg_set_code (coap_msg_t * *msg,* unsigned *code_class,* unsigned *code_detail*)

Set the code in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*code_class*	Code class
`in`	*code_detail*	Code detail

Returns

Operation status

Return values

0	Success
<0	Error

4.3.4.14 int coap_msg_set_msg_id (coap_msg_t * *msg,* unsigned *msg_id*)

Set the message ID in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*msg_id*	Message ID

Returns

Operation status

Return values

0	Success
<0	Error

4.3.4.15 int coap_msg_set_payload (coap_msg_t * *msg,* char * *buf,* size_t *len*)

Set the payload in a message.

Free the buffer in the message structure containing the current payload if there is one, allocate a buffer to contain the new payload and copy the buffer argument into the new payload buffer.

Parameters

`in,out`	*msg*	Pointer to a message structure
`in`	*buf*	Pointer to a buffer containing the payload
`in`	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
<*0*	Error

4.3.4.16 int coap_msg_set_token (coap_msg_t * *msg,* char * *buf,* size_t *len*)

Set the token in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*buf*	Pointer to a buffer containing the token
`in`	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
<*0*	Error

4.3.4.17 int coap_msg_set_type (coap_msg_t * *msg,* unsigned *type*)

Set the type in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*type*	Message type

Returns

Operation status

Return values

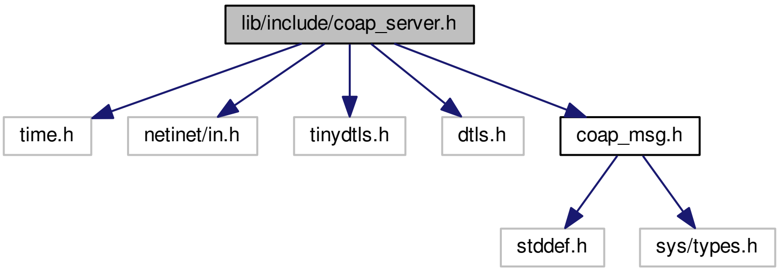
0	Success
<*0*	Error

4.4 lib/include/coap_server.h File Reference

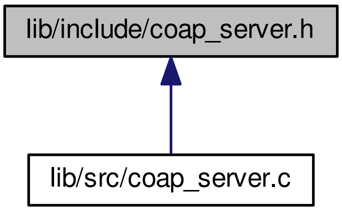
Include file for the FreeCoAP server library.

```
#include <time.h>
#include <netinet/in.h>
#include "tinydtls.h"
#include "dtls.h"
#include "coap_msg.h"
```

Include dependency graph for coap_server.h:



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



Classes

- struct coap_server_path

  URI path structure.
- struct coap_server_path_list_t

  URI path list structure.
- struct coap_server_trans

  Transaction structure.
- struct coap_server

  Server structure.

Macros

- #define COAP_SERVER_NUM_TRANS 8
- #define COAP_SERVER_ADDR_BUF_LEN 128
- #define COAP_SERVER_DIAG_PAYLOAD_LEN 128

Typedefs

- typedef struct coap_server_path coap_server_path_t

  URI path structure.
- typedef struct coap_server_trans coap_server_trans_t

  Transaction structure.
- typedef struct coap_server coap_server_t

  Server structure.

Enumerations

- enum coap_server_resp_t { COAP_SERVER_PIGGYBACKED = 0, COAP_SERVER_SEPARATE = 1 }

  Response type enumeration.
- enum coap_server_dtls_state_t { COAP_SERVER_DTLS_UNCONNECTED = 0, COAP_SERVER_DTLS↩_CONNECTED, COAP_SERVER_DTLS_ALERT }

  Server DTLS state enumeration.

Functions

- int coap_server_create (coap_server_t ∗server, int(∗handle)(coap_server_t ∗, coap_msg_t ∗, coap_msg_t ∗), const char ∗host, const char ∗port, const unsigned char ∗ecdsa_priv_key, const unsigned char ∗ecdsa↩_pub_key_x, const unsigned char ∗ecdsa_pub_key_y)

  Initialise a server structure.
- void coap_server_destroy (coap_server_t ∗server)

  Deinitialise a server structure.
- unsigned coap_server_get_next_msg_id (coap_server_t ∗server)

  Get a new message ID value.
- int coap_server_add_sep_resp_uri_path (coap_server_t ∗server, const char ∗str)

  Register a URI path that requires a separate response.
- int coap_server_run (coap_server_t ∗server)

  Run the server.

4.4.1 Detailed Description

Include file for the FreeCoAP server library.

4.4.2 Macro Definition Documentation

4.4.2.1 #define COAP_SERVER_ADDR_BUF_LEN 128

Buffer length for host addresses

4.4.2.2 #define COAP_SERVER_DIAG_PAYLOAD_LEN 128

Buffer length for diagnostic payloads

4.4.2.3 #define COAP_SERVER_NUM_TRANS 8

Maximum number of active transactions per server

4.4.3 Enumeration Type Documentation

4.4.3.1 enum coap_server_dtls_state_t

Server DTLS state enumeration.

Enumerator

COAP_SERVER_DTLS_UNCONNECTED DTLS session is not active

COAP_SERVER_DTLS_CONNECTED DTLS session is active

COAP_SERVER_DTLS_ALERT A DTLS alert message was received from the client

4.4.3.2 enum coap_server_resp_t

Response type enumeration.

Enumerator

COAP_SERVER_PIGGYBACKED Piggybacked response

COAP_SERVER_SEPARATE Separate response

4.4.4 Function Documentation

4.4.4.1 int coap_server_add_sep_resp_uri_path (coap_server_t ∗ *server,* const char ∗ *str*)

Register a URI path that requires a separate response.

Parameters

in,out	*server*	Pointer to a server structure
in	*str*	String representation of a URI path

Returns

Operation status

Return values

0	Success
<*0*	Error

4.4.4.2 int coap_server_create (coap_server_t ∗ *server,* int(∗)(coap_server_t ∗, coap_msg_t ∗, coap_msg_t ∗) *handle,* const char ∗ *host,* const char ∗ *port,* const unsigned char ∗ *ecdsa_priv_key,* const unsigned char ∗ *ecdsa_pub_key_x,* const unsigned char ∗ *ecdsa_pub_key_y*)

Initialise a server structure.

Parameters

out	*server*	Pointer to a server structure
in	*handle*	Call-back function to handle client requests

in	*host*	Pointer to a string containing the host address of the server
in	*port*	Port number of the server
in	*ecdsa_priv_key*	Buffer containing the ECDSA private key
in	*ecdsa_pub_↩ key_x*	Buffer containing the x component of the ECDSA public key
in	*ecdsa_pub_↩ key_y*	Buffer containing the y component of the ECDSA public key

Returns

Operation status

Return values

0	Success
<0	Error

4.4.4.3 void coap_server_destroy (coap_server_t * *server*)

Deinitialise a server structure.

Parameters

in,out	*server*	Pointer to a server structure

4.4.4.4 unsigned coap_server_get_next_msg_id (coap_server_t * *server*)

Get a new message ID value.

Parameters

in,out	*server*	Pointer to a server structure

Returns

message ID value

4.4.4.5 int coap_server_run (coap_server_t * *server*)

Run the server.

Listen for incoming requests. For each request received, call the handle call-back function in the server structure and send the response to the client.

Parameters

in,out	*server*	Pointer to a server structure

Returns

Operation status

Return values

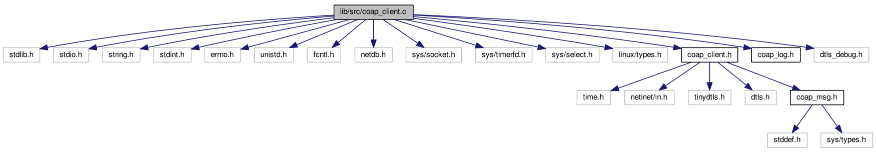
0	Success
<0	Error

4.5 lib/src/coap_client.c File Reference

Source file for the FreeCoAP client library.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/timerfd.h>
#include <sys/select.h>
#include <linux/types.h>
#include "coap_client.h"
#include "coap_log.h"
#include "dtls_debug.h"
```

Include dependency graph for coap_client.c:



Macros

- #define COAP_CLIENT_ACK_TIMEOUT_SEC 2
- #define COAP_CLIENT_MAX_RETRANSMIT 4
- #define COAP_CLIENT_RESP_TIMEOUT_SEC 30
- #define COAP_CLIENT_DTLS_RETRANS_TIMEOUT 100
- #define COAP_CLIENT_DTLS_TOTAL_TIMEOUT 5000

Functions

- int coap_client_create (coap_client_t ∗client, const char ∗host, const char ∗port, const unsigned char ∗ecdsa_priv_key, const unsigned char ∗ecdsa_pub_key_x, const unsigned char ∗ecdsa_pub_key_y)

  Initialise a client structure.

- void coap_client_destroy (coap_client_t ∗client)

  Deinitialise a client structure.

- int coap_client_exchange (coap_client_t ∗client, coap_msg_t ∗req, coap_msg_t ∗resp)

  Send a request to the server and receive the response.

4.5.1 Detailed Description

Source file for the FreeCoAP client library.

4.5.2 Macro Definition Documentation

4.5.2.1 #define COAP_CLIENT_ACK_TIMEOUT_SEC 2

Minimum delay to wait before retransmitting a confirmable message

4.5.2.2 #define COAP_CLIENT_DTLS_RETRANS_TIMEOUT 100

Retransmission timeout (msec) for the DTLS handshake

4.5.2.3 #define COAP_CLIENT_DTLS_TOTAL_TIMEOUT 5000

Total timeout (msec) for the DTLS handshake DTLS priorities

4.5.2.4 #define COAP_CLIENT_MAX_RETRANSMIT 4

Maximum number of times a confirmable message can be retransmitted

4.5.2.5 #define COAP_CLIENT_RESP_TIMEOUT_SEC 30

Maximum amount of time to wait for a response

4.5.3 Function Documentation

4.5.3.1 int coap_client_create (coap_client_t ∗ *client,* const char ∗ *host,* const char ∗ *port,* const unsigned char ∗ *ecdsa_priv_key,* const unsigned char ∗ *ecdsa_pub_key_x,* const unsigned char ∗ *ecdsa_pub_key_y*)

Initialise a client structure.

Parameters

out	*client*	Pointer to a client structure
in	*host*	Pointer to a string containing the host address of the server
in	*port*	Port number of the server
in	*ecdsa_priv_key*	Buffer containing the ECDSA private key
in	*ecdsa_pub_↩ key_x*	Buffer containing the x component of the ECDSA public key
in	*ecdsa_pub_↩ key_y*	Buffer containing the y component of the ECDSA public key

Returns

Operation status

Return values

0	Success
<0	Error

4.5.3.2 void coap_client_destroy (**coap_client_t** ∗ *client*)

Deinitialise a client structure.

Parameters

`in,out`	*client*	Pointer to a client structure

4.5.3.3 int coap_client_exchange (**coap_client_t** ∗ *client,* **coap_msg_t** ∗ *req,* **coap_msg_t** ∗ *resp*)

Send a request to the server and receive the response.

Parameters

`in,out`	*client*	Pointer to a client structure
`in`	*req*	Pointer to the request message
`out`	*resp*	Pointer to the response message

This function sets the message ID and token fields of the request message overriding any values set by the calling function.

Returns

Operation status

Return values

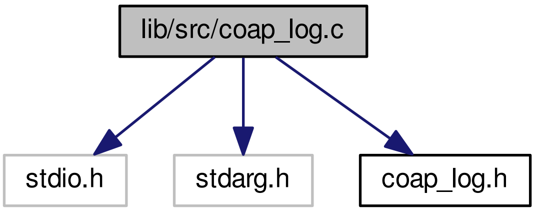
0	Success
<0	Error

4.6 lib/src/coap_log.c File Reference

Source file for the FreeCoAP logging module.

```
#include <stdio.h>
#include <stdarg.h>
#include "coap_log.h"
```

Include dependency graph for coap_log.c:



Functions

- void coap_log_set_level (coap_log_level_t level)

  Set the log level.
- coap_log_level_t coap_log_get_level (void)

  Get the log level.
- void coap_log_error (const char ∗msg,...)

  Log an error message.
- void coap_log_warn (const char ∗msg,...)

  Log a warning message.
- void coap_log_notice (const char ∗msg,...)

  Log an notice message.
- void coap_log_info (const char ∗msg,...)

  Log an info message.
- void coap_log_debug (const char ∗msg,...)

  Log a debug message.

4.6.1 Detailed Description

Source file for the FreeCoAP logging module.

4.6.2 Function Documentation

4.6.2.1 void coap_log_debug (const char ∗ *msg,* ...)

Log a debug message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.6.2.2 void coap_log_error (const char ∗ *msg,* ...)

Log an error message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.6.2.3 coap_log_level_t coap_log_get_level (void)

Get the log level.

Returns

The current log level

4.6.2.4 void coap_log_info (const char ∗ *msg,* ...)

Log an info message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.6.2.5 void coap_log_notice (const char * *msg,* ...)

Log an notice message.

Parameters

in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.6.2.6 void coap_log_set_level (coap_log_level_t *level*)

Set the log level.

Messages with a severity below this level will be filtered. Error messages cannot be filtered.

Parameters

in	*level*	The new log level

< Warning log level

< Notice warning level

< Informational warning level

< Debug warning level

4.6.2.7 void coap_log_warn (const char * *msg,* ...)

Log a warning message.

Parameters

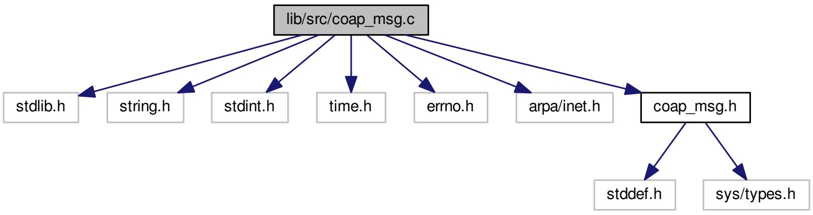
in	*msg*	String containing format specifiers
in	...	arguments for the format specifiers

4.7 lib/src/coap_msg.c File Reference

Source file for the FreeCoAP message parser/formatter library.

```
#include <stdlib.h>
#include <string.h>
#include <stdint.h>
#include <time.h>
#include <errno.h>
#include <arpa/inet.h>
#include "coap_msg.h"
```

Include dependency graph for coap_msg.c:



Macros

- #define coap_msg_op_list_get_first(list) ((list)->first)
- #define coap_msg_op_list_get_last(list) ((list)->last)
- #define coap_msg_op_list_is_empty(list) ((list)->first == NULL)

Functions

- void coap_msg_gen_rand_str (char ∗buf, size_t len)

  Generate a random string of bytes.
- int coap_msg_op_num_is_recognized (unsigned num)

  Check if option is recognized.
- void coap_msg_create (coap_msg_t ∗msg)

  Initialise a message structure.
- void coap_msg_destroy (coap_msg_t ∗msg)

  Deinitialise a message structure.
- void coap_msg_reset (coap_msg_t ∗msg)

  Deinitialise and initialise a message structure.
- unsigned coap_msg_check_critical_ops (coap_msg_t ∗msg)

  Check that all of the critical options in a message are recognized.
- unsigned coap_msg_check_unsafe_ops (coap_msg_t ∗msg)

  Check that all of the unsafe options in a message are recognized.
- int coap_msg_parse_type_msg_id (char ∗buf, size_t len, unsigned ∗type, unsigned ∗msg_id)

  Extract the type and message ID values from a message.
- ssize_t coap_msg_parse (coap_msg_t ∗msg, char ∗buf, size_t len)

  Parse a message.
- int coap_msg_set_type (coap_msg_t ∗msg, unsigned type)

  Set the type in a message.
- int coap_msg_set_code (coap_msg_t ∗msg, unsigned code_class, unsigned code_detail)

  Set the code in a message.
- int coap_msg_set_msg_id (coap_msg_t ∗msg, unsigned msg_id)

  Set the message ID in a message.
- int coap_msg_set_token (coap_msg_t ∗msg, char ∗buf, size_t len)

  Set the token in a message.
- int coap_msg_add_op (coap_msg_t ∗msg, unsigned num, unsigned len, const char ∗val)

  Add a token to a message structure.
- int coap_msg_set_payload (coap_msg_t ∗msg, char ∗buf, size_t len)

  Set the payload in a message.

- ssize_t coap_msg_format (coap_msg_t ∗msg, char ∗buf, size_t len)

  Format a message.
- int coap_msg_copy (coap_msg_t ∗dst, coap_msg_t ∗src)

  Copy a message.

4.7.1 Detailed Description

Source file for the FreeCoAP message parser/formatter library.

4.7.2 Macro Definition Documentation

4.7.2.1 #define coap_msg_op_list_get_first(*list*) ((list)->first)

Get the first option from an option linked-list

4.7.2.2 #define coap_msg_op_list_get_last(*list*) ((list)->last)

Get the last option in an option linked-list

4.7.2.3 #define coap_msg_op_list_is_empty(*list*) ((list)->first == NULL)

Indicate whether or not an option linked-list is empty

4.7.3 Function Documentation

4.7.3.1 int coap_msg_add_op (coap_msg_t ∗ *msg,* unsigned *num,* unsigned *len,* const char ∗ *val*)

Add a token to a message structure.

Parameters

in,out	*msg*	Pointer to a message structure
in	*num*	Option number
in	*len*	Option length
in	*val*	Pointer to a buffer containing the option value

Returns

Operation status

Return values

0	Success
<0	Error

4.7.3.2 unsigned coap_msg_check_critical_ops (coap_msg_t ∗ *msg*)

Check that all of the critical options in a message are recognized.

Parameters

in	*msg*	Pointer to message structure

Returns

Operation status or bad option number

Return values

0	Success
>0	Bad option number

4.7.3.3 unsigned coap_msg_check_unsafe_ops (coap_msg_t * *msg*)

Check that all of the unsafe options in a message are recognized.

Parameters

in	*msg*	Pointer to message structure

Returns

Operation status or bad option number

Return values

0	Success
>0	Bad option number

4.7.3.4 int coap_msg_copy (coap_msg_t * *dst,* coap_msg_t * *src*)

Copy a message.

Parameters

in,out	*dst*	Pointer to the destination message structure
in	*src*	Pointer to the source message structure

Returns

Operation status

Return values

0	Success
<0	Error

4.7.3.5 void coap_msg_create (coap_msg_t * *msg*)

Initialise a message structure.

Parameters

out	*msg*	Pointer to a message structure

4.7.3.6 void coap_msg_destroy (coap_msg_t ∗ *msg*)

Deinitialise a message structure.

Parameters

in,out	*msg*	Pointer to a message structure

4.7.3.7 ssize_t coap_msg_format (coap_msg_t ∗ *msg,* char ∗ *buf,* size_t *len*)

Format a message.

Parameters

in	*msg*	Pointer to a message structure
out	*buf*	Pointer to a buffer to contain the formatted message
in	*len*	Length of the buffer

Returns

Length of the formatted message or error code

Return values

$>$*0*	Length of the formatted message
$<$*0*	Error

4.7.3.8 void coap_msg_gen_rand_str (char ∗ *buf,* size_t *len*)

Generate a random string of bytes.

Parameters

out	*buf*	Pointer to the buffer to store the random string
in	*len*	Length of the buffer

4.7.3.9 int coap_msg_op_num_is_recognized (unsigned *num*)

Check if option is recognized.

Parameters

in	*num*	Option number

Returns

Operation status

Return values

1	Option is recognized
0	Option is not recognized

4.7.3.10 ssize_t coap_msg_parse (coap_msg_t ∗ *msg*, char ∗ *buf*, size_t *len*)

Parse a message.

Parameters

`in,out`	*msg*	Pointer to a message structure
`in`	*buf*	Pointer to a buffer containing the message
`in`	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
<*0*	Error

4.7.3.11 int coap_msg_parse_type_msg_id (char ∗ *buf*, size_t *len*, unsigned ∗ *type*, unsigned ∗ *msg_id*)

Extract the type and message ID values from a message.

If a message contains a format error, this function will attempt to extract the type and message ID so that a reset message can be returned to the sender.

Parameters

`in`	*buf*	Pointer to a buffer containing the message
`in`	*len*	Length of the buffer
`out`	*type*	Pointer to field to store the type value
`out`	*msg_id*	Pointer to a field to store the message ID value

Returns

Operation status

Return values

0	Success
<*0*	Error

4.7.3.12 void coap_msg_reset (coap_msg_t ∗ *msg*)

Deinitialise and initialise a message structure.

Parameters

in,out	*msg*	Pointer to a message structure

4.7.3.13 int coap_msg_set_code (coap_msg_t * *msg,* unsigned *code_class,* unsigned *code_detail*)

Set the code in a message.

Parameters

out	*msg*	Pointer to a message structure
in	*code_class*	Code class
in	*code_detail*	Code detail

Returns

Operation status

Return values

0	Success
<*0*	Error

4.7.3.14 int coap_msg_set_msg_id (coap_msg_t * *msg,* unsigned *msg_id*)

Set the message ID in a message.

Parameters

out	*msg*	Pointer to a message structure
in	*msg_id*	Message ID

Returns

Operation status

Return values

0	Success
<*0*	Error

4.7.3.15 int coap_msg_set_payload (coap_msg_t * *msg,* char * *buf,* size_t *len*)

Set the payload in a message.

Free the buffer in the message structure containing the current payload if there is one, allocate a buffer to contain the new payload and copy the buffer argument into the new payload buffer.

Parameters

in,out	*msg*	Pointer to a message structure
in	*buf*	Pointer to a buffer containing the payload
in	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
$<$*0*	Error

4.7.3.16 int coap_msg_set_token (coap_msg_t ∗ *msg,* char ∗ *buf,* size_t *len*)

Set the token in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*buf*	Pointer to a buffer containing the token
`in`	*len*	Length of the buffer

Returns

Operation status

Return values

0	Success
$<$*0*	Error

4.7.3.17 int coap_msg_set_type (coap_msg_t ∗ *msg,* unsigned *type*)

Set the type in a message.

Parameters

`out`	*msg*	Pointer to a message structure
`in`	*type*	Message type

Returns

Operation status

Return values

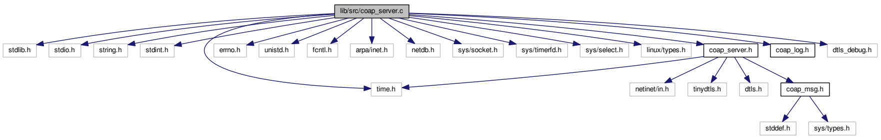
0	Success
$<$*0*	Error

4.8 lib/src/coap_server.c File Reference

Source file for the FreeCoAP server library.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <time.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/timerfd.h>
#include <sys/select.h>
#include <linux/types.h>
#include "coap_server.h"
#include "coap_log.h"
#include "dtls_debug.h"
```

Include dependency graph for coap_server.c:



Macros

- #define COAP_SERVER_ACK_TIMEOUT_SEC 2
- #define COAP_SERVER_MAX_RETRANSMIT 4
- #define COAP_SERVER_DTLS_MTU COAP_MSG_MAX_BUF_LEN
- #define COAP_SERVER_DTLS_RETRANS_TIMEOUT 100
- #define COAP_SERVER_DTLS_TOTAL_TIMEOUT 5000
- #define COAP_SERVER_DTLS_NUM_DH_BITS 1024
- #define COAP_SERVER_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDENCE"

Functions

- int coap_server_create (coap_server_t ∗server, int(∗handle)(coap_server_t ∗, coap_msg_t ∗, coap_msg_t ∗), const char ∗host, const char ∗port, const unsigned char ∗ecdsa_priv_key, const unsigned char ∗ecdsa_pub_key_x, const unsigned char ∗ecdsa_pub_key_y)

  Initialise a server structure.
- void coap_server_destroy (coap_server_t ∗server)

  Deinitialise a server structure.
- unsigned coap_server_get_next_msg_id (coap_server_t ∗server)

  Get a new message ID value.
- int coap_server_add_sep_resp_uri_path (coap_server_t ∗server, const char ∗str)

  Register a URI path that requires a separate response.
- int coap_server_run (coap_server_t ∗server)

  Run the server.

4.8.1 Detailed Description

Source file for the FreeCoAP server library.

4.8.2 Macro Definition Documentation

4.8.2.1 #define COAP_SERVER_ACK_TIMEOUT_SEC 2

Minimum delay to wait before retransmitting a confirmable message

4.8.2.2 #define COAP_SERVER_DTLS_MTU COAP_MSG_MAX_BUF_LEN

Maximum transmission unit excluding the UDP and IPv6 headers

4.8.2.3 #define COAP_SERVER_DTLS_NUM_DH_BITS 1024

DTLS Diffie-Hellman key size

4.8.2.4 #define COAP_SERVER_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDE↩NCE"

DTLS priorities

4.8.2.5 #define COAP_SERVER_DTLS_RETRANS_TIMEOUT 100

Retransmission timeout (msec) for the DTLS handshake

4.8.2.6 #define COAP_SERVER_DTLS_TOTAL_TIMEOUT 5000

Total timeout (msec) for the DTLS handshake

4.8.2.7 #define COAP_SERVER_MAX_RETRANSMIT 4

Maximum number of times a confirmable message can be retransmitted

4.8.3 Function Documentation

4.8.3.1 int coap_server_add_sep_resp_uri_path (coap_server_t * *server,* const char * *str*)

Register a URI path that requires a separate response.

Parameters

`in,out`	*server*	Pointer to a server structure
`in`	*str*	String representation of a URI path

Returns

Operation status

Return values

0	Success
<0	Error

4.8.3.2 int coap_server_create (coap_server_t ∗ *server,* int(∗)(coap_server_t ∗, coap_msg_t ∗, coap_msg_t ∗) *handle,* const char ∗ *host,* const char ∗ *port,* const unsigned char ∗ *ecdsa_priv_key,* const unsigned char ∗ *ecdsa_pub_key_x,* const unsigned char ∗ *ecdsa_pub_key_y*)

Initialise a server structure.

Parameters

`out`	*server*	Pointer to a server structure
`in`	*handle*	Call-back function to handle client requests
`in`	*host*	Pointer to a string containing the host address of the server
`in`	*port*	Port number of the server
`in`	*ecdsa_priv_key*	Buffer containing the ECDSA private key
`in`	*ecdsa_pub_↩ key_x*	Buffer containing the x component of the ECDSA public key
`in`	*ecdsa_pub_↩ key_y*	Buffer containing the y component of the ECDSA public key

Returns

Operation status

Return values

0	Success
<0	Error

4.8.3.3 void coap_server_destroy (coap_server_t ∗ *server*)

Deinitialise a server structure.

Parameters

`in,out`	*server*	Pointer to a server structure

4.8.3.4 unsigned coap_server_get_next_msg_id (coap_server_t ∗ *server*)

Get a new message ID value.

Parameters

`in,out`	*server*	Pointer to a server structure

Returns

message ID value

4.8.3.5 int coap_server_run (coap_server_t ∗ *server*)

Run the server.

Listen for incoming requests. For each request received, call the handle call-back function in the server structure and send the response to the client.

Parameters

in,out	*server*	Pointer to a server structure

Returns

Operation status

Return values

0	Success
<*0*	Error

Index