

FreeCoAP

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	coap_client_t Struct Reference . . . . .	5
3.1.1	Detailed Description . . . . .	5
3.1.2	Member Data Documentation . . . . .	5
3.1.2.1	cred . . . . .	5
3.1.2.2	num_retrans . . . . .	6
3.1.2.3	priority . . . . .	6
3.1.2.4	sd . . . . .	6
3.1.2.5	server_host . . . . .	6
3.1.2.6	server_port . . . . .	6
3.1.2.7	server_sin . . . . .	6
3.1.2.8	server_sin_len . . . . .	6
3.1.2.9	session . . . . .	6
3.1.2.10	timeout . . . . .	6
3.1.2.11	timer_fd . . . . .	6
3.2	coap_msg_op Struct Reference . . . . .	7
3.2.1	Detailed Description . . . . .	7
3.2.2	Member Data Documentation . . . . .	7

3.2.2.1	len	7
3.2.2.2	next	7
3.2.2.3	num	7
3.2.2.4	val	8
3.3	coap_msg_op_list_t Struct Reference	8
3.3.1	Detailed Description	8
3.3.2	Member Data Documentation	8
3.3.2.1	first	8
3.3.2.2	last	9
3.4	coap_msg_t Struct Reference	9
3.4.1	Detailed Description	10
3.4.2	Member Data Documentation	10
3.4.2.1	code_class	10
3.4.2.2	code_detail	10
3.4.2.3	msg_id	10
3.4.2.4	op_list	10
3.4.2.5	payload	10
3.4.2.6	payload_len	10
3.4.2.7	token	10
3.4.2.8	token_len	10
3.4.2.9	type	10
3.4.2.10	ver	11
3.5	coap_server Struct Reference	11
3.5.1	Detailed Description	12
3.5.2	Member Data Documentation	12
3.5.2.1	cred	12
3.5.2.2	dh_params	12
3.5.2.3	handle	12
3.5.2.4	msg_id	12
3.5.2.5	priority	12

3.5.2.6	<a href="#">sd</a>	12
3.5.2.7	<a href="#">sep_list</a>	12
3.5.2.8	<a href="#">trans</a>	12
3.6	<a href="#">coap_server_path Struct Reference</a>	13
3.6.1	<a href="#">Detailed Description</a>	13
3.6.2	<a href="#">Member Data Documentation</a>	13
3.6.2.1	<a href="#">next</a>	13
3.6.2.2	<a href="#">str</a>	13
3.7	<a href="#">coap_server_path_list_t Struct Reference</a>	14
3.7.1	<a href="#">Detailed Description</a>	14
3.7.2	<a href="#">Member Data Documentation</a>	14
3.7.2.1	<a href="#">first</a>	14
3.7.2.2	<a href="#">last</a>	14
3.8	<a href="#">coap_server_trans Struct Reference</a>	15
3.8.1	<a href="#">Detailed Description</a>	16
3.8.2	<a href="#">Member Data Documentation</a>	16
3.8.2.1	<a href="#">active</a>	16
3.8.2.2	<a href="#">client_addr</a>	16
3.8.2.3	<a href="#">client_sin</a>	16
3.8.2.4	<a href="#">client_sin_len</a>	16
3.8.2.5	<a href="#">last_use</a>	16
3.8.2.6	<a href="#">num_retrans</a>	16
3.8.2.7	<a href="#">req</a>	16
3.8.2.8	<a href="#">resp</a>	16
3.8.2.9	<a href="#">server</a>	16
3.8.2.10	<a href="#">session</a>	17
3.8.2.11	<a href="#">timeout</a>	17
3.8.2.12	<a href="#">timer_fd</a>	17

<b>4 File Documentation</b>	<b>19</b>
4.1 lib/include/coap_client.h File Reference	19
4.1.1 Detailed Description	20
4.1.2 Macro Definition Documentation	20
4.1.2.1 COAP_CLIENT_HOST_BUF_LEN	20
4.1.2.2 COAP_CLIENT_PORT_BUF_LEN	20
4.1.3 Function Documentation	20
4.1.3.1 coap_client_create(coap_client_t *client, const char *host, const char *port, const char *key_file_name, const char *cert_file_name, const char *trust_file_name, const char *crl_file_name, const char *common_name)	20
4.1.3.2 coap_client_destroy(coap_client_t *client)	21
4.1.3.3 coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)	21
4.2 lib/include/coap_ipv.h File Reference	21
4.2.1 Detailed Description	22
4.3 lib/include/coap_log.h File Reference	23
4.3.1 Detailed Description	23
4.3.2 Macro Definition Documentation	24
4.3.2.1 COAP_LOG_DEF_LEVEL	24
4.3.3 Enumeration Type Documentation	24
4.3.3.1 coap_log_level_t	24
4.3.4 Function Documentation	24
4.3.4.1 coap_log_debug(const char *msg,...)	24
4.3.4.2 coap_log_error(const char *msg,...)	24
4.3.4.3 coap_log_get_level(void)	24
4.3.4.4 coap_log_info(const char *msg,...)	25
4.3.4.5 coap_log_notice(const char *msg,...)	25
4.3.4.6 coap_log_set_level(coap_log_level_t level)	25
4.3.4.7 coap_log_warn(const char *msg,...)	25
4.4 lib/include/coap_msg.h File Reference	26
4.4.1 Detailed Description	29
4.4.2 Macro Definition Documentation	29

4.4.2.1	<a href="#">coap_msg_get_code_class</a>	29
4.4.2.2	<a href="#">coap_msg_get_code_detail</a>	29
4.4.2.3	<a href="#">coap_msg_get_first_op</a>	29
4.4.2.4	<a href="#">coap_msg_get_msg_id</a>	29
4.4.2.5	<a href="#">coap_msg_get_payload</a>	30
4.4.2.6	<a href="#">coap_msg_get_payload_len</a>	30
4.4.2.7	<a href="#">coap_msg_get_token</a>	30
4.4.2.8	<a href="#">coap_msg_get_token_len</a>	30
4.4.2.9	<a href="#">coap_msg_get_type</a>	30
4.4.2.10	<a href="#">coap_msg_get_ver</a>	30
4.4.2.11	<a href="#">coap_msg_is_empty</a>	30
4.4.2.12	<a href="#">COAP_MSG_MAX_BUF_LEN</a>	30
4.4.2.13	<a href="#">COAP_MSG_MAX_CODE_CLASS</a>	30
4.4.2.14	<a href="#">COAP_MSG_MAX_CODE_DETAIL</a>	30
4.4.2.15	<a href="#">COAP_MSG_MAX_MSG_ID</a>	31
4.4.2.16	<a href="#">COAP_MSG_MAX_TOKEN_LEN</a>	31
4.4.2.17	<a href="#">coap_msg_op_get_len</a>	31
4.4.2.18	<a href="#">coap_msg_op_get_next</a>	31
4.4.2.19	<a href="#">coap_msg_op_get_num</a>	31
4.4.2.20	<a href="#">coap_msg_op_get_val</a>	31
4.4.2.21	<a href="#">COAP_MSG_OP_MAX_BLOCK_SIZE</a>	31
4.4.2.22	<a href="#">COAP_MSG_OP_MAX_BLOCK_VAL_LEN</a>	31
4.4.2.23	<a href="#">coap_msg_op_num_is_critical</a>	31
4.4.2.24	<a href="#">coap_msg_op_num_is_unsafe</a>	31
4.4.2.25	<a href="#">coap_msg_op_num_no_cache_key</a>	32
4.4.2.26	<a href="#">coap_msg_op_set_len</a>	32
4.4.2.27	<a href="#">coap_msg_op_set_next</a>	32
4.4.2.28	<a href="#">coap_msg_op_set_num</a>	32
4.4.2.29	<a href="#">coap_msg_op_set_val</a>	32
4.4.2.30	<a href="#">COAP_MSG_OP_URI_PATH_MAX_LEN</a>	32

4.4.2.31	COAP_MSG_VER . . . . .	32
4.4.3	Enumeration Type Documentation . . . . .	32
4.4.3.1	coap_msg_class_t . . . . .	32
4.4.3.2	coap_msg_client_err_t . . . . .	33
4.4.3.3	coap_msg_method_t . . . . .	33
4.4.3.4	coap_msg_op_num_t . . . . .	33
4.4.3.5	coap_msg_server_err_t . . . . .	34
4.4.3.6	coap_msg_success_t . . . . .	34
4.4.3.7	coap_msg_type_t . . . . .	34
4.4.4	Function Documentation . . . . .	34
4.4.4.1	coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, const char *val) . . . . .	34
4.4.4.2	coap_msg_check_critical_ops(coap_msg_t *msg) . . . . .	35
4.4.4.3	coap_msg_check_unsafe_ops(coap_msg_t *msg) . . . . .	35
4.4.4.4	coap_msg_copy(coap_msg_t *dst, coap_msg_t *src) . . . . .	36
4.4.4.5	coap_msg_create(coap_msg_t *msg) . . . . .	36
4.4.4.6	coap_msg_destroy(coap_msg_t *msg) . . . . .	36
4.4.4.7	coap_msg_format(coap_msg_t *msg, char *buf, size_t len) . . . . .	36
4.4.4.8	coap_msg_gen_rand_str(char *buf, size_t len) . . . . .	37
4.4.4.9	coap_msg_op_format_block_val(char *val, unsigned len, unsigned num, unsigned more, unsigned size) . . . . .	37
4.4.4.10	coap_msg_op_num_is_recognized(unsigned num) . . . . .	38
4.4.4.11	coap_msg_op_parse_block_val(unsigned *num, unsigned *more, unsigned *size, const char *val, unsigned len) . . . . .	38
4.4.4.12	coap_msg_parse(coap_msg_t *msg, char *buf, size_t len) . . . . .	38
4.4.4.13	coap_msg_parse_type_msg_id(char *buf, size_t len, unsigned *type, unsigned *msg_id) . . . . .	39
4.4.4.14	coap_msg_reset(coap_msg_t *msg) . . . . .	39
4.4.4.15	coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code↵_detail) . . . . .	40
4.4.4.16	coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id) . . . . .	40
4.4.4.17	coap_msg_set_payload(coap_msg_t *msg, char *buf, size_t len) . . . . .	40
4.4.4.18	coap_msg_set_token(coap_msg_t *msg, char *buf, size_t len) . . . . .	41



4.4.4.19	<code>coap_msg_set_type(coap_msg_t *msg, unsigned type)</code>	41
4.4.4.20	<code>coap_msg_uri_path_to_str(coap_msg_t *msg, char *buf, size_t len)</code>	42
4.5	lib/include/coap_server.h File Reference	42
4.5.1	Detailed Description	44
4.5.2	Macro Definition Documentation	44
4.5.2.1	<code>COAP_SERVER_ADDR_BUF_LEN</code>	44
4.5.2.2	<code>COAP_SERVER_DIAG_PAYLOAD_LEN</code>	44
4.5.2.3	<code>COAP_SERVER_NUM_TRANS</code>	44
4.5.3	Enumeration Type Documentation	44
4.5.3.1	<code>coap_server_resp_t</code>	44
4.5.4	Function Documentation	44
4.5.4.1	<code>coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str)</code>	44
4.5.4.2	<code>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 char *key_file_name, const char *cert_file_name, const char *trust_file_name, const char *crl_file_name)</code>	45
4.5.4.3	<code>coap_server_destroy(coap_server_t *server)</code>	45
4.5.4.4	<code>coap_server_get_next_msg_id(coap_server_t *server)</code>	46
4.5.4.5	<code>coap_server_run(coap_server_t *server)</code>	46
4.6	lib/src/coap_client.c File Reference	46
4.6.1	Detailed Description	47
4.6.2	Macro Definition Documentation	48
4.6.2.1	<code>COAP_CLIENT_ACK_TIMEOUT_SEC</code>	48
4.6.2.2	<code>COAP_CLIENT_DTLS_MTU</code>	48
4.6.2.3	<code>COAP_CLIENT_DTLS_PRIORITIES</code>	48
4.6.2.4	<code>COAP_CLIENT_DTLS_RETRANS_TIMEOUT</code>	48
4.6.2.5	<code>COAP_CLIENT_DTLS_TOTAL_TIMEOUT</code>	48
4.6.2.6	<code>COAP_CLIENT_MAX_RETRANSMIT</code>	48
4.6.2.7	<code>COAP_CLIENT_RESP_TIMEOUT_SEC</code>	48
4.6.3	Function Documentation	48
4.6.3.1	<code>coap_client_create(coap_client_t *client, const char *host, const char *port, const char *key_file_name, const char *cert_file_name, const char *trust_file_name, const char *crl_file_name, const char *common_name)</code>	48

4.6.3.2	<code>coap_client_destroy(coap_client_t *client)</code>	49
4.6.3.3	<code>coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)</code>	49
4.7	<code>lib/src/coap_log.c</code> File Reference	49
4.7.1	Detailed Description	50
4.7.2	Function Documentation	50
4.7.2.1	<code>coap_log_debug(const char *msg,...)</code>	50
4.7.2.2	<code>coap_log_error(const char *msg,...)</code>	51
4.7.2.3	<code>coap_log_get_level(void)</code>	51
4.7.2.4	<code>coap_log_info(const char *msg,...)</code>	51
4.7.2.5	<code>coap_log_notice(const char *msg,...)</code>	51
4.7.2.6	<code>coap_log_set_level(coap_log_level_t level)</code>	51
4.7.2.7	<code>coap_log_warn(const char *msg,...)</code>	52
4.8	<code>lib/src/coap_msg.c</code> File Reference	52
4.8.1	Detailed Description	53
4.8.2	Macro Definition Documentation	54
4.8.2.1	<code>coap_msg_op_list_get_first</code>	54
4.8.2.2	<code>coap_msg_op_list_get_last</code>	54
4.8.2.3	<code>coap_msg_op_list_is_empty</code>	54
4.8.3	Function Documentation	54
4.8.3.1	<code>coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, const char *val)</code>	54
4.8.3.2	<code>coap_msg_check_critical_ops(coap_msg_t *msg)</code>	54
4.8.3.3	<code>coap_msg_check_unsafe_ops(coap_msg_t *msg)</code>	55
4.8.3.4	<code>coap_msg_copy(coap_msg_t *dst, coap_msg_t *src)</code>	55
4.8.3.5	<code>coap_msg_create(coap_msg_t *msg)</code>	56
4.8.3.6	<code>coap_msg_destroy(coap_msg_t *msg)</code>	56
4.8.3.7	<code>coap_msg_format(coap_msg_t *msg, char *buf, size_t len)</code>	56
4.8.3.8	<code>coap_msg_gen_rand_str(char *buf, size_t len)</code>	56
4.8.3.9	<code>coap_msg_op_format_block_val(char *val, unsigned len, unsigned num, unsigned more, unsigned size)</code>	57
4.8.3.10	<code>coap_msg_op_num_is_recognized(unsigned num)</code>	57

4.8.3.11	coap_msg_op_parse_block_val(unsigned *num, unsigned *more, unsigned *size, const char *val, unsigned len) . . . . .	57
4.8.3.12	coap_msg_parse(coap_msg_t *msg, char *buf, size_t len) . . . . .	58
4.8.3.13	coap_msg_parse_type_msg_id(char *buf, size_t len, unsigned *type, unsigned *msg_id) . . . . .	58
4.8.3.14	coap_msg_reset(coap_msg_t *msg) . . . . .	59
4.8.3.15	coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code←_detail) . . . . .	59
4.8.3.16	coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id) . . . . .	59
4.8.3.17	coap_msg_set_payload(coap_msg_t *msg, char *buf, size_t len) . . . . .	60
4.8.3.18	coap_msg_set_token(coap_msg_t *msg, char *buf, size_t len) . . . . .	60
4.8.3.19	coap_msg_set_type(coap_msg_t *msg, unsigned type) . . . . .	61
4.8.3.20	coap_msg_uri_path_to_str(coap_msg_t *msg, char *buf, size_t len) . . . . .	61
4.9	lib/src/coap_server.c File Reference . . . . .	61
4.9.1	Detailed Description . . . . .	63
4.9.2	Macro Definition Documentation . . . . .	63
4.9.2.1	COAP_SERVER_ACK_TIMEOUT_SEC . . . . .	63
4.9.2.2	COAP_SERVER_DTLS_MTU . . . . .	63
4.9.2.3	COAP_SERVER_DTLS_NUM_DH_BITS . . . . .	63
4.9.2.4	COAP_SERVER_DTLS_PRIORITIES . . . . .	63
4.9.2.5	COAP_SERVER_DTLS_RETRANS_TIMEOUT . . . . .	63
4.9.2.6	COAP_SERVER_DTLS_TOTAL_TIMEOUT . . . . .	63
4.9.2.7	COAP_SERVER_MAX_RETRANSMIT . . . . .	63
4.9.3	Function Documentation . . . . .	63
4.9.3.1	coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str) . .	63
4.9.3.2	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 char *key_file_name, const char *cert_file_name, const char *trust_file_name, const char *crl_file_name) . . . . .	64
4.9.3.3	coap_server_destroy(coap_server_t *server) . . . . .	64
4.9.3.4	coap_server_get_next_msg_id(coap_server_t *server) . . . . .	65
4.9.3.5	coap_server_run(coap_server_t *server) . . . . .	65



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">coap_client_t</a>		
Client structure	. . . . .	5
<a href="#">coap_msg_op</a>		
Option structure	. . . . .	7
<a href="#">coap_msg_op_list_t</a>		
Option linked-list structure	. . . . .	8
<a href="#">coap_msg_t</a>		
Message structure	. . . . .	9
<a href="#">coap_server</a>		
Server structure	. . . . .	11
<a href="#">coap_server_path</a>		
URI path structure	. . . . .	13
<a href="#">coap_server_path_list_t</a>		
URI path list structure	. . . . .	14
<a href="#">coap_server_trans</a>		
Transaction structure	. . . . .	15



## Chapter 2

# File Index

### 2.1 File List

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

lib/include/ <a href="#">coap_client.h</a>	
Include file for the FreeCoAP client library . . . . .	19
lib/include/ <a href="#">coap_ipv.h</a>	
Include file for the FreeCoAP IP Version (IPv4/IPv6) abstraction layer . . . . .	21
lib/include/ <a href="#">coap_log.h</a>	
Include file for the FreeCoAP logging module . . . . .	23
lib/include/ <a href="#">coap_msg.h</a>	
Include file for the FreeCoAP message parser/formatter library . . . . .	26
lib/include/ <a href="#">coap_server.h</a>	
Include file for the FreeCoAP server library . . . . .	42
lib/src/ <a href="#">coap_client.c</a>	
Source file for the FreeCoAP client library . . . . .	46
lib/src/ <a href="#">coap_log.c</a>	
Source file for the FreeCoAP logging module . . . . .	49
lib/src/ <a href="#">coap_msg.c</a>	
Source file for the FreeCoAP message parser/formatter library . . . . .	52
lib/src/ <a href="#">coap_server.c</a>	
Source file for the FreeCoAP server library . . . . .	61





## 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](#)
- coap\_ipv\_sockaddr\_in\_t [server\\_sin](#)
- socklen\_t [server\\_sin\\_len](#)
- char [server\\_host](#) [COAP\_CLIENT\_HOST\_BUF\_LEN]
- char [server\\_port](#) [COAP\_CLIENT\_PORT\_BUF\_LEN]
- gnutls\_session\_t [session](#)
- gnutls\_certificate\_credentials\_t [cred](#)
- gnutls\_priority\_t [priority](#)

#### 3.1.1 Detailed Description

Client structure.

#### 3.1.2 Member Data Documentation

##### 3.1.2.1 gnutls\_certificate\_credentials\_t coap\_client\_t::cred

DTLS credentials

### 3.1.2.2 unsigned coap\_client\_t::num\_retrans

Current number of retransmissions

### 3.1.2.3 gnutls\_priority\_t coap\_client\_t::priority

DTLS priorities

### 3.1.2.4 int coap\_client\_t::sd

Socket descriptor

### 3.1.2.5 char coap\_client\_t::server\_host[COAP\_CLIENT\_HOST\_BUF\_LEN]

String to hold the server host address

### 3.1.2.6 char coap\_client\_t::server\_port[COAP\_CLIENT\_PORT\_BUF\_LEN]

String to hold the server port number

### 3.1.2.7 coap\_ipvs\_sockaddr\_in\_t coap\_client\_t::server\_sin

Socket structure

### 3.1.2.8 socklen\_t coap\_client\_t::server\_sin\_len

Socket structure length

### 3.1.2.9 gnutls\_session\_t coap\_client\_t::session

DTLS session

### 3.1.2.10 struct timespec coap\_client\_t::timeout

Timeout value

### 3.1.2.11 int coap\_client\_t::timer\_fd

Timer file descriptor

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

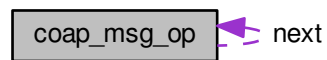
- lib/include/[coap\\_client.h](#)

## 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:

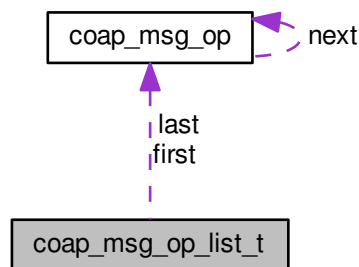
- lib/include/coap\_msg.h

## 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:

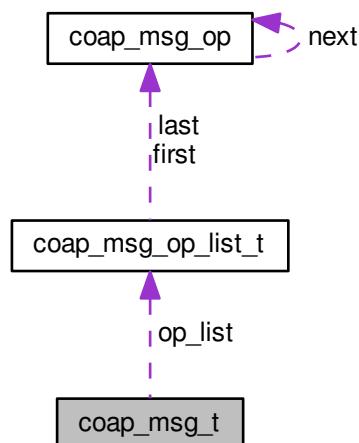
- lib/include/coap\_msg.h

## 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:

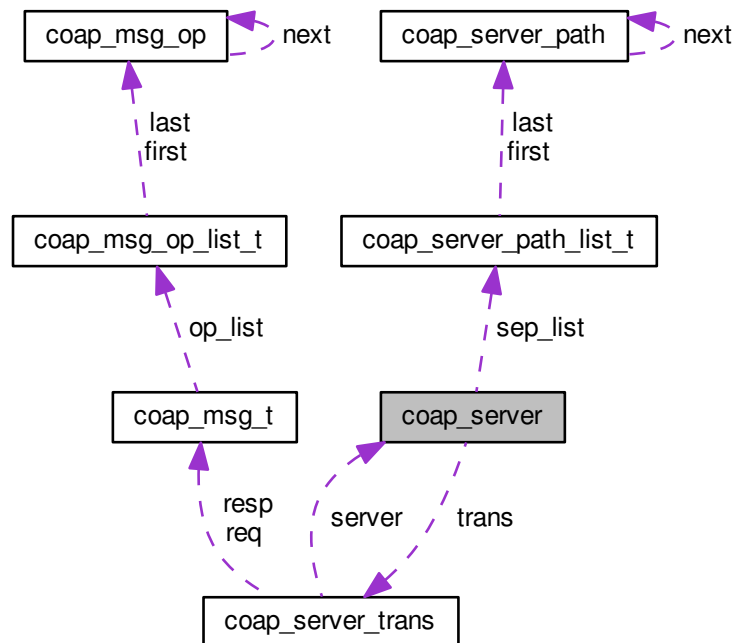
- lib/include/coap\_msg.h

## 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` \*)
- `gnutls_certificate_credentials_t` `cred`
- `gnutls_priority_t` `priority`
- `gnutls_dh_params_t` `dh_params`

### 3.5.1 Detailed Description

Server structure.

### 3.5.2 Member Data Documentation

#### 3.5.2.1 `gnutls_certificate_credentials_t coap_server::cred`

DTLS credentials

#### 3.5.2.2 `gnutls_dh_params_t coap_server::dh_params`

Diffie-Hellman parameters

#### 3.5.2.3 `int(* coap_server::handle)(struct coap_server *, coap_msg_t *, coap_msg_t *)`

Call-back function to handle requests and generate responses

#### 3.5.2.4 `unsigned coap_server::msg_id`

Last message ID value used in a response message

#### 3.5.2.5 `gnutls_priority_t coap_server::priority`

DTLS priorities

#### 3.5.2.6 `int coap_server::sd`

Socket descriptor

#### 3.5.2.7 `coap_server_path_list_t coap_server::sep_list`

List of URI paths that require separate responses

#### 3.5.2.8 `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](#)

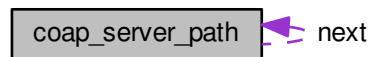


## 3.6 coap\_server\_path Struct Reference

URI path structure.

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

Collaboration diagram for coap\_server\_path:



### 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:

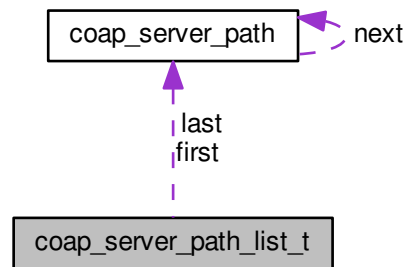
- lib/include/[coap\\_server.h](#)

### 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:

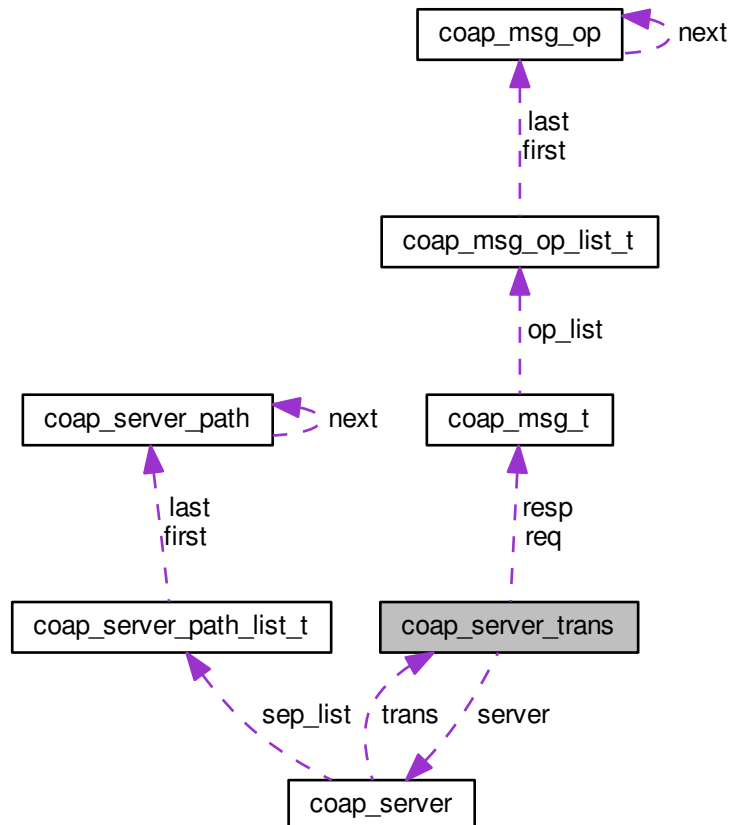
- lib/include/[coap\\_server.h](#)

## 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`
- coap\_ipv\_sockaddr\_in\_t `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`
- gnutls\_session\_t `session`

### 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 `char coap_server_trans::client_addr[COAP_SERVER_ADDR_BUF_LEN]`

String to hold the client address

#### 3.8.2.3 `coap_ipv_sockaddr_in_t coap_server_trans::client_sin`

Socket structure

#### 3.8.2.4 `socklen_t coap_server_trans::client_sin_len`

Socket structure length

#### 3.8.2.5 `time_t coap_server_trans::last_use`

The time that this transaction structure was last used

#### 3.8.2.6 `unsigned coap_server_trans::num_retrans`

Current number of retransmissions

#### 3.8.2.7 `coap_msg_t coap_server_trans::req`

Last request message received for this transaction

#### 3.8.2.8 `coap_msg_t coap_server_trans::resp`

Last response message sent for this transaction

#### 3.8.2.9 `struct coap_server* coap_server_trans::server`

Pointer to the containing server structure

#### 3.8.2.10 gnutls\_session\_t coap\_server\_trans::session

DTLS session

#### 3.8.2.11 struct timespec coap\_server\_trans::timeout

Timeout value

#### 3.8.2.12 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

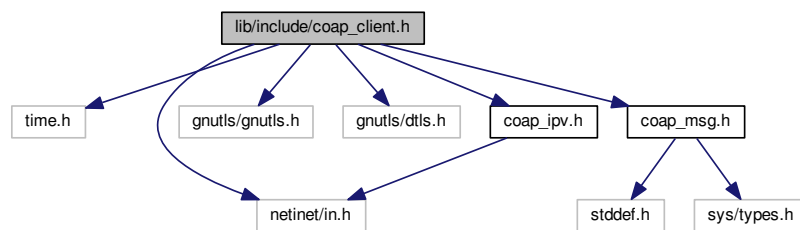
# File Documentation

### 4.1 lib/include/coap\_client.h File Reference

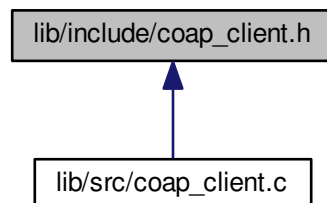
Include file for the FreeCoAP client library.

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

## Functions

- int [coap\\_client\\_create](#) ([coap\\_client\\_t](#) \*client, const char \*host, const char \*port, const char \*key\_file\_name, const char \*cert\_file\_name, const char \*trust\_file\_name, const char \*crl\_file\_name, const char \*common\_name)  
*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\_HOST\_BUF\_LEN 128

Buffer length for host addresses

#### 4.1.2.2 #define COAP\_CLIENT\_PORT\_BUF\_LEN 8

Buffer length for port numbers

### 4.1.3 Function Documentation

#### 4.1.3.1 int coap\_client\_create ( coap\_client\_t \* client, const char \* host, const char \* port, const char \* key\_file\_name, const char \* cert\_file\_name, const char \* trust\_file\_name, const char \* crl\_file\_name, const char \* common\_name )

Initialise a client structure.

#### Parameters

out	<i>client</i>	Pointer to a client structure
in	<i>host</i>	Pointer to a string containing the host address of the server
in	<i>port</i>	Port number of the server
in	<i>key_file_name</i>	String containing the DTLS key file name
in	<i>cert_file_name</i>	String containing the DTLS certificate file name
in	<i>trust_file_name</i>	String containing the DTLS trust file name



**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.1.3.2 void coap\_client\_destroy ( coap\_client\_t \* client )**

Deinitialise a client structure.

**Parameters**

in, out	<i>client</i>	Pointer to a client structure
---------	---------------	-------------------------------

**4.1.3.3 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.

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

**Parameters**

in, out	<i>client</i>	Pointer to a client structure
in	<i>req</i>	Pointer to the request message
out	<i>resp</i>	Pointer to the response message

**Returns**

Operation status

**Return values**

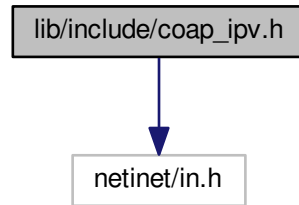
0	Success
<0	Error

## 4.2 lib/include/coap\_ipv.h File Reference

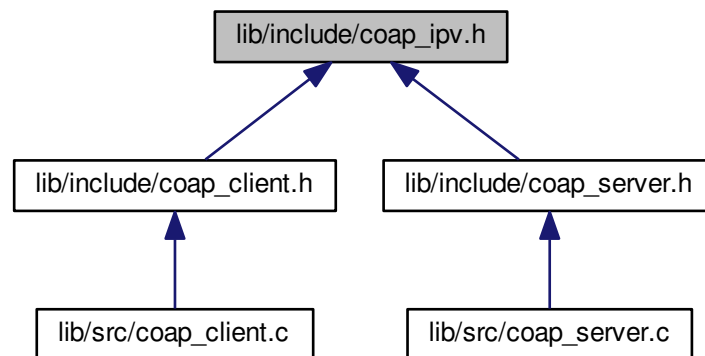
Include file for the FreeCoAP IP Version (IPv4/IPv6) abstraction layer.

```
#include <netinet/in.h>
```

Include dependency graph for coap\_ipv.h:



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



## Macros

- `#define COAP_IPV_AF_INET AF_INET`
- `#define COAP_IPV_SIN_ADDR sin_addr`
- `#define COAP_IPV_SIN_PORT sin_port`

## Typedefs

- `typedef struct sockaddr_in coap_ipv_sockaddr_in_t`

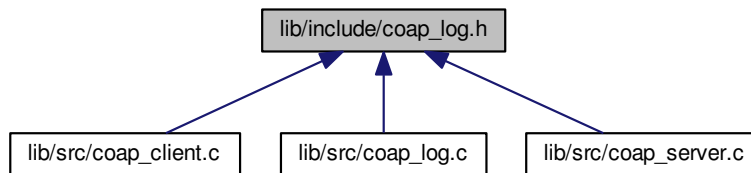
### 4.2.1 Detailed Description

Include file for the FreeCoAP IP Version (IPv4/IPv6) abstraction layer.

## 4.3 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.*

#### 4.3.1 Detailed Description

Include file for the FreeCoAP logging module.

## 4.3.2 Macro Definition Documentation

### 4.3.2.1 #define COAP\_LOG\_DEF\_LEVEL COAP\_LOG\_ERROR

Default log level

## 4.3.3 Enumeration Type Documentation

### 4.3.3.1 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

## 4.3.4 Function Documentation

### 4.3.4.1 void coap\_log\_debug ( const char \* *msg*, ... )

Log a debug message.

Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

### 4.3.4.2 void coap\_log\_error ( const char \* *msg*, ... )

Log an error message.

Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

### 4.3.4.3 coap\_log\_level\_t coap\_log\_get\_level ( void )

Get the log level.

### Returns

The current log level

#### 4.3.4.4 void coap\_log\_info ( const char \* *msg*, ... )

Log an info message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.3.4.5 void coap\_log\_notice ( const char \* *msg*, ... )

Log an notice message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.3.4.6 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	<i>level</i>	The new log level
----	--------------	-------------------

< Warning log level

< Notice warning level

< Informational warning level

< Debug warning level

#### 4.3.4.7 void coap\_log\_warn ( const char \* *msg*, ... )

Log a warning message.

## Parameters

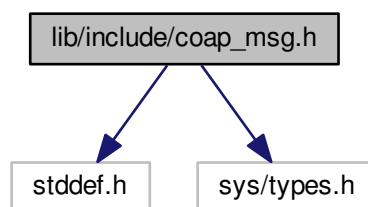
in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

## 4.4 lib/include/coap\_msg.h File Reference

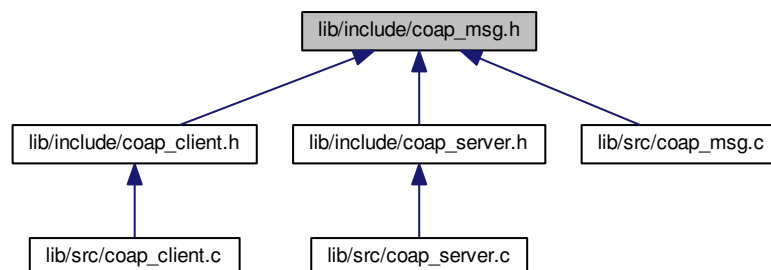
Include file for the FreeCoAP message parser/formatter library.

```
#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\\_MAX\\_LEN](#) 256
- #define [COAP\\_MSG\\_OP\\_MAX\\_BLOCK\\_VAL\\_LEN](#) 3
- #define [COAP\\_MSG\\_OP\\_MAX\\_BLOCK\\_SIZE](#) (1 << 10)
- #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 }

*Request code detail enumeration.*

- enum `coap_msg_success_t` {  
`COAP_MSG_CREATED` = 1, `COAP_MSG_DELETED` = 2, `COAP_MSG_VALID` = 3, `COAP_MSG_CHANNEL_CLOSED` = 4,  
`COAP_MSG_CONTENT` = 5, `COAP_MSG_CONTINUE` = 31 }  
*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_INCOMPLETE` = 8,  
`COAP_MSG_PRECOND_FAILED` = 12, `COAP_MSG_REQ_ENT_TOO_LARGE` = 13, `COAP_MSG_UNSUPPORTED_FORMAT` = 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_BLOCK2` = 23, `COAP_MSG_BLOCK1` = 27, `COAP_MSG_SIZE2` = 28, `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.*
- int `coap_msg_op_parse_block_val` (unsigned \*num, unsigned \*more, unsigned \*size, const char \*val, unsigned len)  
*Parse Block1 or Block2 option value.*
- int `coap_msg_op_format_block_val` (char \*val, unsigned len, unsigned num, unsigned more, unsigned size)  
*Format Block1 or Block2 option value.*
- 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.*
- size\_t [coap\\_msg\\_uri\\_path\\_to\\_str](#) (coap\_msg\_t \*msg, char \*buf, size\_t len)  
*Convert the URI path in a message to a string representation.*

#### 4.4.1 Detailed Description

Include file for the FreeCoAP message parser/formatter library.

#### 4.4.2 Macro Definition Documentation

4.4.2.1 `#define coap_msg_get_code_class( msg ) ((msg)->code_class)`

Get the code class from a message

4.4.2.2 `#define coap_msg_get_code_detail( msg ) ((msg)->code_detail)`

Get the code detail from a message

4.4.2.3 `#define coap_msg_get_first_op( msg ) ((msg)->op_list.first)`

Get the first option from a message

4.4.2.4 `#define coap_msg_get_msg_id( msg ) ((msg)->msg_id)`

Get the message ID from message

4.4.2.5 `#define coap_msg_get_payload( msg ) ((msg)->payload)`

Get the payload from a message

4.4.2.6 `#define coap_msg_get_payload_len( msg ) ((msg)->payload_len)`

Get the payload length from a message

4.4.2.7 `#define coap_msg_get_token( msg ) ((msg)->token)`

Get the token from a message

4.4.2.8 `#define coap_msg_get_token_len( msg ) ((msg)->token_len)`

Get the token length from a message

4.4.2.9 `#define coap_msg_get_type( msg ) ((msg)->type)`

Get the type from a message

4.4.2.10 `#define coap_msg_get_ver( msg ) ((msg)->ver)`

Get the version from a message

4.4.2.11 `#define coap_msg_is_empty( msg ) (((msg)->code_class == 0) && ((msg)->code_detail == 0))`

Indicate if a message is empty

4.4.2.12 `#define COAP_MSG_MAX_BUF_LEN 1152`

Maximum buffer length for header and payload

4.4.2.13 `#define COAP_MSG_MAX_CODE_CLASS 7`

Maximum code class

4.4.2.14 `#define COAP_MSG_MAX_CODE_DETAIL 31`

Maximum code detail

4.4.2.15 `#define COAP_MSG_MAX_MSG_ID ((1 << 16) - 1)`

Maximum message ID

4.4.2.16 `#define COAP_MSG_MAX_TOKEN_LEN 8`

Maximum token length

4.4.2.17 `#define coap_msg_op_get_len( op ) ((op)->len)`

Get the option length from an option

4.4.2.18 `#define coap_msg_op_get_next( op ) ((op)->next)`

Get the next pointer from an option

4.4.2.19 `#define coap_msg_op_get_num( op ) ((op)->num)`

Get the option number from an option

4.4.2.20 `#define coap_msg_op_get_val( op ) ((op)->val)`

Get the option value from an option

4.4.2.21 `#define COAP_MSG_OP_MAX_BLOCK_SIZE (1 << 10)`

Maximum block size for a Block1 or Block2 option

4.4.2.22 `#define COAP_MSG_OP_MAX_BLOCK_VAL_LEN 3`

Maximum buffer length for a Block1 or Block2 option value

4.4.2.23 `#define coap_msg_op_num_is_critical( num ) ((num) & 1)`

Indicate if an option is critical

4.4.2.24 `#define coap_msg_op_num_is_unsafe( num ) ((num) & 2)`

Indicate if an option is unsafe to forward

4.4.2.25 `#define coap_msg_op_num_no_cache_key( num ) ((num & 0x1e) == 0x1c)`

Indicate if an option is not part of the cache key

4.4.2.26 `#define coap_msg_op_set_len( op, len ) ((op)->len = (len))`

Set the option length in an option

4.4.2.27 `#define coap_msg_op_set_next( op, next_op ) ((op)->next = (next_op))`

Set the next pointer in an option

4.4.2.28 `#define coap_msg_op_set_num( op, num ) ((op)->num = (num))`

Set the option number in an option

4.4.2.29 `#define coap_msg_op_set_val( op, val ) ((op)->val = (val))`

Set the option value in an option

4.4.2.30 `#define COAP_MSG_OP_URI_PATH_MAX_LEN 256`

Maximum buffer length for a reconstructed URI path

4.4.2.31 `#define COAP_MSG_VER 0x01`

CoAP version

### 4.4.3 Enumeration Type Documentation

4.4.3.1 `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

## 4.4.3.2 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\_INCOMPLETE** Request entity incomplete 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

## 4.4.3.3 enum coap\_msg\_method\_t

Request 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

## 4.4.3.4 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\_BLOCK2** Block2 option number  
**COAP\_MSG\_BLOCK1** Block1 option number  
**COAP\_MSG\_SIZE2** Size2 option number  
**COAP\_MSG\_PROXY\_URI** Proxy-URI option number  
**COAP\_MSG\_PROXY\_SCHEME** Proxy-Scheme option number  
**COAP\_MSG\_SIZE1** Size1 option number

#### 4.4.3.5 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

#### 4.4.3.6 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  
**COAP\_MSG\_CONTINUE** Continue success response

#### 4.4.3.7 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

### 4.4.4 Function Documentation

#### 4.4.4.1 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	<i>msg</i>	Pointer to a message structure
in	<i>num</i>	Option number
in	<i>len</i>	Option length
in	<i>val</i>	Pointer to a buffer containing the option value

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.4.4.2 unsigned coap\_msg\_check\_critical\_ops ( coap\_msg\_t \* msg )**

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

**Parameters**

in	<i>msg</i>	Pointer to message structure
----	------------	------------------------------

**Returns**

Operation status or bad option number

**Return values**

0	Success
>0	Bad option number

**4.4.4.3 unsigned coap\_msg\_check\_unsafe\_ops ( coap\_msg\_t \* msg )**

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

**Parameters**

in	<i>msg</i>	Pointer to message structure
----	------------	------------------------------

**Returns**

Operation status or bad option number

**Return values**

0	Success
>0	Bad option number

**4.4.4.4 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

**4.4.4.5 void coap\_msg\_create ( coap\_msg\_t \* msg )**

Initialise a message structure.

**Parameters**

out	msg	Pointer to a message structure
-----	-----	--------------------------------

**4.4.4.6 void coap\_msg\_destroy ( coap\_msg\_t \* msg )**

Deinitialise a message structure.

**Parameters**

in, out	msg	Pointer to a message structure
---------	-----	--------------------------------

**4.4.4.7 ssize\_t coap\_msg\_format ( coap\_msg\_t \* msg, char \* buf, size\_t len )**

Format a message.



## Parameters

in	<i>msg</i>	Pointer to a message structure
out	<i>buf</i>	Pointer to a buffer to contain the formatted message
in	<i>len</i>	Length of the buffer

## Returns

Length of the formatted message or error code

## Return values

>0	Length of the formatted message
<0	Error

**4.4.4.8 void coap\_msg\_gen\_rand\_str ( char \* *buf*, size\_t *len* )**

Generate a random string of bytes.

## Parameters

out	<i>buf</i>	Pointer to the buffer to store the random string
in	<i>len</i>	Length of the buffer

**4.4.4.9 int coap\_msg\_op\_format\_block\_val ( char \* *val*, unsigned *len*, unsigned *num*, unsigned *more*, unsigned *size* )**

Format Block1 or Block2 option value.

## Parameters

out	<i>val</i>	Pointer to option value
in	<i>len</i>	Length of option value
in	<i>num</i>	Block number
in	<i>more</i>	More value
in	<i>size</i>	Block size

## Returns

Length of the formatted option value or error code

## Return values

>0	Length of the formatted option value
<0	Error

#### 4.4.4.10 int coap\_msg\_op\_num\_is\_recognized ( unsigned num )

Check if option is recognized.

##### Parameters

in	<i>num</i>	Option number
----	------------	---------------

##### Returns

Operation status

##### Return values

1	Option is recognized
0	Option is not recognized

#### 4.4.4.11 int coap\_msg\_op\_parse\_block\_val ( unsigned \* num, unsigned \* more, unsigned \* size, const char \* val, unsigned len )

Parse Block1 or Block2 option value.

##### Parameters

out	<i>num</i>	Pointer to Block number
out	<i>more</i>	Pointer to More value
out	<i>size</i>	Pointer to Block size
in	<i>val</i>	Pointer to the option value
in	<i>len</i>	Option length

##### Returns

Operation status

##### Return values

0	Success
<0	Error

#### 4.4.4.12 ssize\_t coap\_msg\_parse ( coap\_msg\_t \* msg, char \* buf, size\_t len )

Parse a message.

##### Parameters

in, out	<i>msg</i>	Pointer to a message structure
---------	------------	--------------------------------

## Parameters

in	<i>buf</i>	Pointer to a buffer containing the message
in	<i>len</i>	Length of the buffer

## Returns

Operation status

## Return values

0	Success
<0	Error

#### 4.4.4.13 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	<i>buf</i>	Pointer to a buffer containing the message
in	<i>len</i>	Length of the buffer
out	<i>type</i>	Pointer to field to store the type value
out	<i>msg_id</i>	Pointer to a field to store the message ID value

## Returns

Operation status

## Return values

0	Success
<0	Error

#### 4.4.4.14 void coap\_msg\_reset ( coap\_msg\_t \* *msg* )

Deinitialise and initialise a message structure.

## Parameters

in, out	<i>msg</i>	Pointer to a message structure
---------	------------	--------------------------------

#### 4.4.4.15 int coap\_msg\_set\_code ( coap\_msg\_t \* msg, unsigned code\_class, unsigned code\_detail )

Set the code in a message.

##### Parameters

out	<i>msg</i>	Pointer to a message structure
in	<i>code_class</i>	Code class
in	<i>code_detail</i>	Code detail

##### Returns

Operation status

##### Return values

0	Success
<0	Error

#### 4.4.4.16 int coap\_msg\_set\_msg\_id ( coap\_msg\_t \* msg, unsigned msg\_id )

Set the message ID in a message.

##### Parameters

out	<i>msg</i>	Pointer to a message structure
in	<i>msg</i> ↔ <i>_id</i>	Message ID

##### Returns

Operation status

##### Return values

0	Success
<0	Error

#### 4.4.4.17 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	<i>msg</i>	Pointer to a message structure
in	<i>buf</i>	Pointer to a buffer containing the payload
in	<i>len</i>	Length of the buffer

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.4.4.18** int coap\_msg\_set\_token ( coap\_msg\_t \* *msg*, char \* *buf*, size\_t *len* )

Set the token in a message.

## Parameters

out	<i>msg</i>	Pointer to a message structure
in	<i>buf</i>	Pointer to a buffer containing the token
in	<i>len</i>	Length of the buffer

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.4.4.19** int coap\_msg\_set\_type ( coap\_msg\_t \* *msg*, unsigned *type* )

Set the type in a message.

## Parameters

out	<i>msg</i>	Pointer to a message structure
in	<i>type</i>	Message type

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.4.4.20** `size_t coap_msg_uri_path_to_str ( coap_msg_t * msg, char * buf, size_t len )`

Convert the URI path in a message to a string representation.

**Parameters**

in	<i>msg</i>	Pointer to a message structure
out	<i>buf</i>	Pointer to a buffer to hold the string
in	<i>len</i>	Length of the buffer

**Returns**

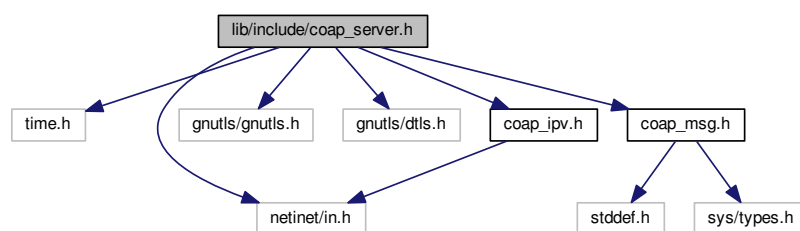
The number of bytes that would be written to the buffer it was large enough

## 4.5 lib/include/coap\_server.h File Reference

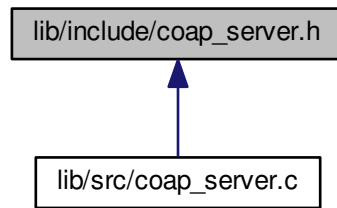
Include file for the FreeCoAP server library.

```
#include <time.h>
#include <netinet/in.h>
#include <gnutls/gnutls.h>
#include <gnutls/dtls.h>
#include "coap_msg.h"
#include "coap_ipv.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.*

## 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 char \*key\_file\_name, const char \*cert\_file\_name, const char \*trust\_file\_name, const char \*crl\_file\_name)  
*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.*

### 4.5.1 Detailed Description

Include file for the FreeCoAP server library.

### 4.5.2 Macro Definition Documentation

#### 4.5.2.1 #define COAP\_SERVER\_ADDR\_BUF\_LEN 128

Buffer length for host addresses

#### 4.5.2.2 #define COAP\_SERVER\_DIAG\_PAYLOAD\_LEN 128

Buffer length for diagnostic payloads

#### 4.5.2.3 #define COAP\_SERVER\_NUM\_TRANS 8

Maximum number of active transactions per server

### 4.5.3 Enumeration Type Documentation

#### 4.5.3.1 enum coap\_server\_resp\_t

Response type enumeration.

Enumerator

**COAP\_SERVER\_PIGGYBACKED** Piggybacked response  
**COAP\_SERVER\_SEPARATE** Separate response

### 4.5.4 Function Documentation

#### 4.5.4.1 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	<i>server</i>	Pointer to a server structure
in	<i>str</i>	String representation of a URI path

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.5.4.2** `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 char * key_file_name, const char * cert_file_name, const char * trust_file_name, const char * crl_file_name )`

Initialise a server structure.

## Parameters

out	<i>server</i>	Pointer to a server structure
in	<i>handle</i>	Call-back function to handle client requests
in	<i>host</i>	Pointer to a string containing the host address of the server
in	<i>port</i>	Port number of the server
in	<i>key_file_name</i>	String containing the DTLS key file name
in	<i>cert_file_name</i>	String containing the DTLS certificate file name
in	<i>trust_file_name</i>	String containing the DTLS trust file name
in	<i>crl_file_name</i>	String containing the DTLS certificate revocation list file name

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.5.4.3** `void coap_server_destroy ( coap_server_t * server )`

Deinitialise a server structure.

## Parameters

in, out	<i>server</i>	Pointer to a server structure
---------	---------------	-------------------------------

#### 4.5.4.4 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

#### 4.5.4.5 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

## 4.6 lib/src/coap\_client.c File Reference

Source file for the FreeCoAP client library.



## 4.6.2 Macro Definition Documentation

### 4.6.2.1 #define COAP\_CLIENT\_ACK\_TIMEOUT\_SEC 2

Minimum delay to wait before retransmitting a confirmable message

### 4.6.2.2 #define COAP\_CLIENT\_DTLS\_MTU COAP\_MSG\_MAX\_BUF\_LEN

Maximum transmission unit excluding the UDP and IPv6 headers

### 4.6.2.3 #define COAP\_CLIENT\_DTLS\_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER\_PRECEDENCE"

DTLS priorities

### 4.6.2.4 #define COAP\_CLIENT\_DTLS\_RETRANS\_TIMEOUT 100

Retransmission timeout (msec) for the DTLS handshake

### 4.6.2.5 #define COAP\_CLIENT\_DTLS\_TOTAL\_TIMEOUT 5000

Total timeout (msec) for the DTLS handshake

### 4.6.2.6 #define COAP\_CLIENT\_MAX\_RETRANSMIT 4

Maximum number of times a confirmable message can be retransmitted

### 4.6.2.7 #define COAP\_CLIENT\_RESP\_TIMEOUT\_SEC 30

Maximum amount of time to wait for a response

## 4.6.3 Function Documentation

### 4.6.3.1 int coap\_client\_create ( coap\_client\_t \* client, const char \* host, const char \* port, const char \* key\_file\_name, const char \* cert\_file\_name, const char \* trust\_file\_name, const char \* crl\_file\_name, const char \* common\_name )

Initialise a client structure.

#### Parameters

out	<i>client</i>	Pointer to a client structure
in	<i>host</i>	Pointer to a string containing the host address of the server
in	<i>port</i>	Port number of the server
in	<i>key_file_name</i>	String containing the DTLS key file name
in	<i>cert_file_name</i>	String containing the DTLS certificate file name
in	<i>trust_file_name</i>	String containing the DTLS trust file name
in	<i>crls_file_name</i>	String containing the DTLS certificate revocation list file name
in	<i>common_name</i>	String containing the common name of the server

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.6.3.2 void coap\_client\_destroy ( coap\_client\_t \* client )**

Deinitialise a client structure.

**Parameters**

in, out	<i>client</i>	Pointer to a client structure
---------	---------------	-------------------------------

**4.6.3.3 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.

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

**Parameters**

in, out	<i>client</i>	Pointer to a client structure
in	<i>req</i>	Pointer to the request message
out	<i>resp</i>	Pointer to the response message

**Returns**

Operation status

**Return values**

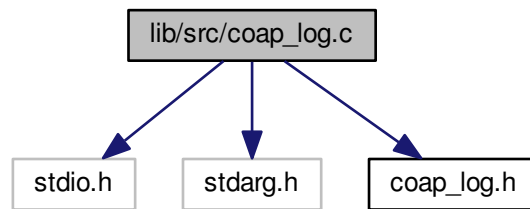
0	Success
<0	Error

**4.7 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.*

### 4.7.1 Detailed Description

Source file for the FreeCoAP logging module.

### 4.7.2 Function Documentation

#### 4.7.2.1 void [coap\\_log\\_debug](#) ( const char \* *msg*, ... )

Log a debug message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.7.2.2 void coap\_log\_error ( const char \* *msg*, ... )

Log an error message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.7.2.3 coap\_log\_level\_t coap\_log\_get\_level ( void )

Get the log level.

##### Returns

The current log level

#### 4.7.2.4 void coap\_log\_info ( const char \* *msg*, ... )

Log an info message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.7.2.5 void coap\_log\_notice ( const char \* *msg*, ... )

Log an notice message.

##### Parameters

in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

#### 4.7.2.6 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	<i>level</i>	The new log level
----	--------------	-------------------

< Warning log level

< Notice warning level

< Informational warning level

< Debug warning level

#### 4.7.2.7 void coap\_log\_warn ( const char \* *msg*, ... )

Log a warning message.

##### Parameters

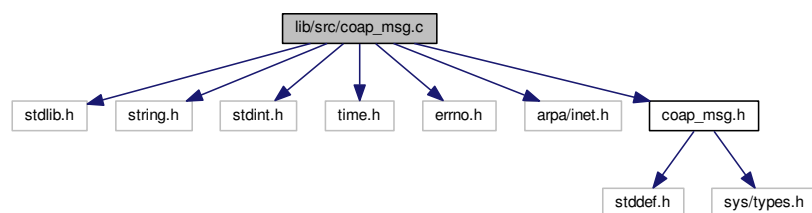
in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

## 4.8 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.*
- int [coap\\_msg\\_op\\_parse\\_block\\_val](#) (unsigned \*num, unsigned \*more, unsigned \*size, const char \*val, unsigned len)  
*Parse Block1 or Block2 option value.*
- int [coap\\_msg\\_op\\_format\\_block\\_val](#) (char \*val, unsigned len, unsigned num, unsigned more, unsigned size)  
*Format Block1 or Block2 option value.*
- 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.*
- size\_t [coap\\_msg\\_uri\\_path\\_to\\_str](#) (coap\_msg\_t \*msg, char \*buf, size\_t len)  
*Convert the URI path in a message to a string representation.*

### 4.8.1 Detailed Description

Source file for the FreeCoAP message parser/formatter library.

## 4.8.2 Macro Definition Documentation

### 4.8.2.1 `#define coap_msg_op_list_get_first( list ) ((list)->first)`

Get the first option from an option linked-list

### 4.8.2.2 `#define coap_msg_op_list_get_last( list ) ((list)->last)`

Get the last option in an option linked-list

### 4.8.2.3 `#define coap_msg_op_list_is_empty( list ) ((list)->first == NULL)`

Indicate whether or not an option linked-list is empty

## 4.8.3 Function Documentation

### 4.8.3.1 `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	<i>msg</i>	Pointer to a message structure
in	<i>num</i>	Option number
in	<i>len</i>	Option length
in	<i>val</i>	Pointer to a buffer containing the option value

#### Returns

Operation status

#### Return values

0	Success
<0	Error

### 4.8.3.2 `unsigned coap_msg_check_critical_ops ( coap_msg_t * msg )`

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

#### Parameters

in	<i>msg</i>	Pointer to message structure
----	------------	------------------------------

**Returns**

Operation status or bad option number

**Return values**

0	Success
>0	Bad option number

**4.8.3.3 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

**4.8.3.4 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

#### 4.8.3.5 void coap\_msg\_create ( coap\_msg\_t \* msg )

Initialise a message structure.

##### Parameters

out	<i>msg</i>	Pointer to a message structure
-----	------------	--------------------------------

#### 4.8.3.6 void coap\_msg\_destroy ( coap\_msg\_t \* msg )

Deinitialise a message structure.

##### Parameters

in, out	<i>msg</i>	Pointer to a message structure
---------	------------	--------------------------------

#### 4.8.3.7 ssize\_t coap\_msg\_format ( coap\_msg\_t \* msg, char \* buf, size\_t len )

Format a message.

##### Parameters

in	<i>msg</i>	Pointer to a message structure
out	<i>buf</i>	Pointer to a buffer to contain the formatted message
in	<i>len</i>	Length of the buffer

##### Returns

Length of the formatted message or error code

##### Return values

>0	Length of the formatted message
<0	Error

#### 4.8.3.8 void coap\_msg\_gen\_rand\_str ( char \* buf, size\_t len )

Generate a random string of bytes.

##### Parameters

out	<i>buf</i>	Pointer to the buffer to store the random string
in	<i>len</i>	Length of the buffer

4.8.3.9 `int coap_msg_op_format_block_val ( char * val, unsigned len, unsigned num, unsigned more, unsigned size )`

Format Block1 or Block2 option value.

#### Parameters

out	<i>val</i>	Pointer to option value
in	<i>len</i>	Length of option value
in	<i>num</i>	Block number
in	<i>more</i>	More value
in	<i>size</i>	Block size

#### Returns

Length of the formatted option value or error code

#### Return values

>0	Length of the formatted option value
<0	Error

4.8.3.10 `int coap_msg_op_num_is_recognized ( unsigned num )`

Check if option is recognized.

#### Parameters

in	<i>num</i>	Option number
----	------------	---------------

#### Returns

Operation status

#### Return values

1	Option is recognized
0	Option is not recognized

4.8.3.11 `int coap_msg_op_parse_block_val ( unsigned * num, unsigned * more, unsigned * size, const char * val, unsigned len )`

Parse Block1 or Block2 option value.

#### Parameters

out	<i>num</i>	Pointer to Block number
-----	------------	-------------------------

**Parameters**

out	<i>more</i>	Pointer to More value
out	<i>size</i>	Pointer to Block size
in	<i>val</i>	Pointer to the option value
in	<i>len</i>	Option length

**Returns**

Operation status

**Return values**

0	Success
<0	Error

#### 4.8.3.12 `ssize_t coap_msg_parse ( coap_msg_t * msg, char * buf, size_t len )`

Parse a message.

**Parameters**

in, out	<i>msg</i>	Pointer to a message structure
in	<i>buf</i>	Pointer to a buffer containing the message
in	<i>len</i>	Length of the buffer

**Returns**

Operation status

**Return values**

0	Success
<0	Error

#### 4.8.3.13 `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	<i>buf</i>	Pointer to a buffer containing the message
in	<i>len</i>	Length of the buffer

## Parameters

out	<i>type</i>	Pointer to field to store the type value
out	<i>msg</i> ↔ <i>_id</i>	Pointer to a field to store the message ID value

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.8.3.14 void coap\_msg\_reset ( coap\_msg\_t \* msg )**

Deinitialise and initialise a message structure.

## Parameters

in, out	<i>msg</i>	Pointer to a message structure
---------	------------	--------------------------------

**4.8.3.15 int coap\_msg\_set\_code ( coap\_msg\_t \* msg, unsigned code\_class, unsigned code\_detail )**

Set the code in a message.

## Parameters

out	<i>msg</i>	Pointer to a message structure
in	<i>code_class</i>	Code class
in	<i>code_detail</i>	Code detail

## Returns

Operation status

## Return values

0	Success
<0	Error

**4.8.3.16 int coap\_msg\_set\_msg\_id ( coap\_msg\_t \* msg, unsigned msg\_id )**

Set the message ID in a message.

**Parameters**

out	<i>msg</i>	Pointer to a message structure
in	<i>msg</i> ↔ <i>_id</i>	Message ID

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.8.3.17 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	<i>msg</i>	Pointer to a message structure
in	<i>buf</i>	Pointer to a buffer containing the payload
in	<i>len</i>	Length of the buffer

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.8.3.18 int coap\_msg\_set\_token ( coap\_msg\_t \* msg, char \* buf, size\_t len )**

Set the token in a message.

**Parameters**

out	<i>msg</i>	Pointer to a message structure
in	<i>buf</i>	Pointer to a buffer containing the token
in	<i>len</i>	Length of the buffer



**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.8.3.19 int coap\_msg\_set\_type ( coap\_msg\_t \* msg, unsigned type )**

Set the type in a message.

**Parameters**

out	<i>msg</i>	Pointer to a message structure
in	<i>type</i>	Message type

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.8.3.20 size\_t coap\_msg\_uri\_path\_to\_str ( coap\_msg\_t \* msg, char \* buf, size\_t len )**

Convert the URI path in a message to a string representation.

**Parameters**

in	<i>msg</i>	Pointer to a message structure
out	<i>buf</i>	Pointer to a buffer to hold the string
in	<i>len</i>	Length of the buffer

**Returns**

The number of bytes that would be written to the buffer it was large enough

**4.9 lib/src/coap\_server.c File Reference**

Source file for the FreeCoAP server library.



### 4.9.1 Detailed Description

Source file for the FreeCoAP server library.

### 4.9.2 Macro Definition Documentation

#### 4.9.2.1 `#define COAP_SERVER_ACK_TIMEOUT_SEC 2`

Minimum delay to wait before retransmitting a confirmable message

#### 4.9.2.2 `#define COAP_SERVER_DTLS_MTU COAP_MSG_MAX_BUF_LEN`

Maximum transmission unit excluding the UDP and IPv6 headers

#### 4.9.2.3 `#define COAP_SERVER_DTLS_NUM_DH_BITS 1024`

DTLS Diffie-Hellman key size

#### 4.9.2.4 `#define COAP_SERVER_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDE↵NCE"`

DTLS priorities

#### 4.9.2.5 `#define COAP_SERVER_DTLS_RETRANS_TIMEOUT 100`

Retransmission timeout (msec) for the DTLS handshake

#### 4.9.2.6 `#define COAP_SERVER_DTLS_TOTAL_TIMEOUT 5000`

Total timeout (msec) for the DTLS handshake

#### 4.9.2.7 `#define COAP_SERVER_MAX_RETRANSMIT 4`

Maximum number of times a confirmable message can be retransmitted

### 4.9.3 Function Documentation

#### 4.9.3.1 `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	<i>server</i>	Pointer to a server structure
in	<i>str</i>	String representation of a URI path

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.9.3.2** `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 char * key_file_name, const char * cert_file_name, const char * trust_file_name, const char * crl_file_name )`

Initialise a server structure.

**Parameters**

out	<i>server</i>	Pointer to a server structure
in	<i>handle</i>	Call-back function to handle client requests
in	<i>host</i>	Pointer to a string containing the host address of the server
in	<i>port</i>	Port number of the server
in	<i>key_file_name</i>	String containing the DTLS key file name
in	<i>cert_file_name</i>	String containing the DTLS certificate file name
in	<i>trust_file_name</i>	String containing the DTLS trust file name
in	<i>crl_file_name</i>	String containing the DTLS certificate revocation list file name

**Returns**

Operation status

**Return values**

0	Success
<0	Error

**4.9.3.3** `void coap_server_destroy ( coap_server_t * server )`

Deinitialise a server structure.

**Parameters**

in, out	<i>server</i>	Pointer to a server structure
---------	---------------	-------------------------------

#### 4.9.3.4 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

#### 4.9.3.5 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

active

coap\_server\_trans, [16](#)

COAP\_CLIENT\_ACK\_TIMEOUT\_SEC

coap\_client.c, [48](#)

COAP\_CLIENT\_DTLS\_MTU

coap\_client.c, [48](#)

COAP\_CLIENT\_DTLS\_PRIORITIES

coap\_client.c, [48](#)

COAP\_CLIENT\_DTLS\_RETRANS\_TIMEOUT

coap\_client.c, [48](#)

COAP\_CLIENT\_DTLS\_TOTAL\_TIMEOUT

coap\_client.c, [48](#)

COAP\_CLIENT\_HOST\_BUF\_LEN

coap\_client.h, [20](#)

COAP\_CLIENT\_MAX\_RETRANSMIT

coap\_client.c, [48](#)

COAP\_CLIENT\_PORT\_BUF\_LEN

coap\_client.h, [20](#)

COAP\_CLIENT\_RESP\_TIMEOUT\_SEC

coap\_client.c, [48](#)

COAP\_LOG\_DEBUG

coap\_log.h, [24](#)

COAP\_LOG\_DEF\_LEVEL

coap\_log.h, [24](#)

COAP\_LOG\_ERROR

coap\_log.h, [24](#)

COAP\_LOG\_INFO

coap\_log.h, [24](#)

COAP\_LOG\_NOTICE

coap\_log.h, [24](#)

COAP\_LOG\_WARN

coap\_log.h, [24](#)

COAP\_MSG\_ACCEPT

coap\_msg.h, [33](#)

COAP\_MSG\_ACK

coap\_msg.h, [34](#)

COAP\_MSG\_BAD\_GATEWAY

coap\_msg.h, [34](#)

COAP\_MSG\_BAD\_OPTION

coap\_msg.h, [33](#)

COAP\_MSG\_BAD\_REQ

coap\_msg.h, [33](#)

COAP\_MSG\_BLOCK1

coap\_msg.h, [33](#)

COAP\_MSG\_BLOCK2

coap\_msg.h, [33](#)

COAP\_MSG\_CHANGED

coap\_msg.h, [34](#)

COAP\_MSG\_CLIENT\_ERR

coap\_msg.h, [32](#)

COAP\_MSG\_CONTENT\_FORMAT

coap\_msg.h, [33](#)

COAP\_MSG\_CONTENT

coap\_msg.h, [34](#)

COAP\_MSG\_CONTINUE

coap\_msg.h, [34](#)

COAP\_MSG\_CON

coap\_msg.h, [34](#)

COAP\_MSG\_CREATED

coap\_msg.h, [34](#)

COAP\_MSG\_DELETED

coap\_msg.h, [34](#)

COAP\_MSG\_DELETE

coap\_msg.h, [33](#)

COAP\_MSG\_ETAG

coap\_msg.h, [33](#)

COAP\_MSG\_FORBIDDEN

coap\_msg.h, [33](#)

COAP\_MSG\_GATEWAY\_TIMEOUT

coap\_msg.h, [34](#)

COAP\_MSG\_GET

coap\_msg.h, [33](#)

COAP\_MSG\_IF\_MATCH

coap\_msg.h, [33](#)

COAP\_MSG\_IF\_NONE\_MATCH

coap\_msg.h, [33](#)

COAP\_MSG\_INCOMPLETE

coap\_msg.h, [33](#)

COAP\_MSG\_INT\_SERVER\_ERR

coap\_msg.h, [34](#)

COAP\_MSG\_LOCATION\_PATH

coap\_msg.h, [33](#)

COAP\_MSG\_LOCATION\_QUERY

coap\_msg.h, [33](#)

COAP\_MSG\_MAX\_AGE

coap\_msg.h, [33](#)

COAP\_MSG\_MAX\_BUF\_LEN

coap\_msg.h, [30](#)

COAP\_MSG\_MAX\_CODE\_CLASS

coap\_msg.h, [30](#)

COAP\_MSG\_MAX\_CODE\_DETAIL

coap\_msg.h, [30](#)

COAP\_MSG\_MAX\_MSG\_ID

coap\_msg.h, [30](#)

COAP\_MSG\_MAX\_TOKEN\_LEN

coap\_msg.h, [31](#)

COAP\_MSG\_METHOD\_NOT\_ALLOWED

coap\_msg.h, [33](#)

COAP\_MSG\_NOT\_ACCEPTABLE  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_NOT\_FOUND  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_NOT\_IMPL  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_NON  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_OP\_MAX\_BLOCK\_SIZE  
     coap\_msg.h, [31](#)  
 COAP\_MSG\_OP\_MAX\_BLOCK\_VAL\_LEN  
     coap\_msg.h, [31](#)  
 COAP\_MSG\_OP\_URI\_PATH\_MAX\_LEN  
     coap\_msg.h, [32](#)  
 COAP\_MSG\_POST  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_PRECOND\_FAILED  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_PROXY\_NOT\_SUP  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_PROXY\_SCHEME  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_PROXY\_URI  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_PUT  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_REQ\_ENT\_TOO\_LARGE  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_REQ  
     coap\_msg.h, [32](#)  
 COAP\_MSG\_RST  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_SERV\_UNAVAIL  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_SERVER\_ERR  
     coap\_msg.h, [32](#)  
 COAP\_MSG\_SIZE1  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_SIZE2  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_SUCCESS  
     coap\_msg.h, [32](#)  
 COAP\_MSG\_UNAUTHORIZED  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_UNSUP\_CONT\_FMT  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_URI\_HOST  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_URI\_PATH  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_URI\_PORT  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_URI\_QUERY  
     coap\_msg.h, [33](#)  
 COAP\_MSG\_VALID  
     coap\_msg.h, [34](#)  
 COAP\_MSG\_VER  
     coap\_msg.h, [32](#)  
 COAP\_SERVER\_ACK\_TIMEOUT\_SEC  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_ADDR\_BUF\_LEN  
     coap\_server.h, [44](#)  
 COAP\_SERVER\_DIAG\_PAYLOAD\_LEN  
     coap\_server.h, [44](#)  
 COAP\_SERVER\_DTLS\_MTU  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_DTLS\_NUM\_DH\_BITS  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_DTLS\_PRIORITIES  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_DTLS\_RETRANS\_TIMEOUT  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_DTLS\_TOTAL\_TIMEOUT  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_MAX\_RETRANSMIT  
     coap\_server.c, [63](#)  
 COAP\_SERVER\_NUM\_TRANS  
     coap\_server.h, [44](#)  
 COAP\_SERVER\_PIGGYBACKED  
     coap\_server.h, [44](#)  
 COAP\_SERVER\_SEPARATE  
     coap\_server.h, [44](#)  
 client\_addr  
     coap\_server\_trans, [16](#)  
 client\_sin  
     coap\_server\_trans, [16](#)  
 client\_sin\_len  
     coap\_server\_trans, [16](#)  
 coap\_client.c  
     COAP\_CLIENT\_ACK\_TIMEOUT\_SEC, [48](#)  
     COAP\_CLIENT\_DTLS\_MTU, [48](#)  
     COAP\_CLIENT\_DTLS\_PRIORITIES, [48](#)  
     COAP\_CLIENT\_DTLS\_RETRANS\_TIMEOUT, [48](#)  
     COAP\_CLIENT\_DTLS\_TOTAL\_TIMEOUT, [48](#)  
     COAP\_CLIENT\_MAX\_RETRANSMIT, [48](#)  
     COAP\_CLIENT\_RESP\_TIMEOUT\_SEC, [48](#)  
     coap\_client\_create, [48](#)  
     coap\_client\_destroy, [49](#)  
     coap\_client\_exchange, [49](#)  
 coap\_client.h  
     COAP\_CLIENT\_HOST\_BUF\_LEN, [20](#)  
     COAP\_CLIENT\_PORT\_BUF\_LEN, [20](#)  
     coap\_client\_create, [20](#)  
     coap\_client\_destroy, [21](#)  
     coap\_client\_exchange, [21](#)  
 coap\_client\_create  
     coap\_client.c, [48](#)  
     coap\_client.h, [20](#)  
 coap\_client\_destroy  
     coap\_client.c, [49](#)  
     coap\_client.h, [21](#)  
 coap\_client\_exchange  
     coap\_client.c, [49](#)  
     coap\_client.h, [21](#)  
 coap\_client\_t, [5](#)  
     cred, [5](#)



- num\_retrans, [5](#)
- priority, [6](#)
- sd, [6](#)
- server\_host, [6](#)
- server\_port, [6](#)
- server\_sin, [6](#)
- server\_sin\_len, [6](#)
- session, [6](#)
- timeout, [6](#)
- timer\_fd, [6](#)
- coap\_log.c
  - coap\_log\_debug, [50](#)
  - coap\_log\_error, [50](#)
  - coap\_log\_get\_level, [51](#)
  - coap\_log\_info, [51](#)
  - coap\_log\_notice, [51](#)
  - coap\_log\_set\_level, [51](#)
  - coap\_log\_warn, [52](#)
- coap\_log.h
  - COAP\_LOG\_DEBUG, [24](#)
  - COAP\_LOG\_DEF\_LEVEL, [24](#)
  - COAP\_LOG\_ERROR, [24](#)
  - COAP\_LOG\_INFO, [24](#)
  - COAP\_LOG\_NOTICE, [24](#)
  - COAP\_LOG\_WARN, [24](#)
  - coap\_log\_debug, [24](#)
  - coap\_log\_error, [24](#)
  - coap\_log\_get\_level, [24](#)
  - coap\_log\_info, [25](#)
  - coap\_log\_level\_t, [24](#)
  - coap\_log\_notice, [25](#)
  - coap\_log\_set\_level, [25](#)
  - coap\_log\_warn, [25](#)
- coap\_log\_debug
  - coap\_log.c, [50](#)
  - coap\_log.h, [24](#)
- coap\_log\_error
  - coap\_log.c, [50](#)
  - coap\_log.h, [24](#)
- coap\_log\_get\_level
  - coap\_log.c, [51](#)
  - coap\_log.h, [24](#)
- coap\_log\_info
  - coap\_log.c, [51](#)
  - coap\_log.h, [25](#)
- coap\_log\_level\_t
  - coap\_log.h, [24](#)
- coap\_log\_notice
  - coap\_log.c, [51](#)
  - coap\_log.h, [25](#)
- coap\_log\_set\_level
  - coap\_log.c, [51](#)
  - coap\_log.h, [25](#)
- coap\_log\_warn
  - coap\_log.c, [52](#)
  - coap\_log.h, [25](#)
- coap\_msg.c
  - coap\_msg\_add\_op, [54](#)
  - coap\_msg\_check\_critical\_ops, [54](#)
  - coap\_msg\_check\_unsafe\_ops, [55](#)
  - coap\_msg\_copy, [55](#)
  - coap\_msg\_create, [55](#)
  - coap\_msg\_destroy, [56](#)
  - coap\_msg\_format, [56](#)
  - coap\_msg\_gen\_rand\_str, [56](#)
  - coap\_msg\_op\_format\_block\_val, [56](#)
  - coap\_msg\_op\_list\_get\_first, [54](#)
  - coap\_msg\_op\_list\_get\_last, [54](#)
  - coap\_msg\_op\_list\_is\_empty, [54](#)
  - coap\_msg\_op\_num\_is\_recognized, [57](#)
  - coap\_msg\_op\_parse\_block\_val, [57](#)
  - coap\_msg\_parse, [58](#)
  - coap\_msg\_parse\_type\_msg\_id, [58](#)
  - coap\_msg\_reset, [59](#)
  - coap\_msg\_set\_code, [59](#)
  - coap\_msg\_set\_msg\_id, [59](#)
  - coap\_msg\_set\_payload, [60](#)
  - coap\_msg\_set\_token, [60](#)
  - coap\_msg\_set\_type, [61](#)
  - coap\_msg\_uri\_path\_to\_str, [61](#)
- coap\_msg.h
  - COAP\_MSG\_ACCEPT, [33](#)
  - COAP\_MSG\_ACK, [34](#)
  - COAP\_MSG\_BAD\_GATEWAY, [34](#)
  - COAP\_MSG\_BAD\_OPTION, [33](#)
  - COAP\_MSG\_BAD\_REQ, [33](#)
  - COAP\_MSG\_BLOCK1, [33](#)
  - COAP\_MSG\_BLOCK2, [33](#)
  - COAP\_MSG\_CHANGED, [34](#)
  - COAP\_MSG\_CLIENT\_ERR, [32](#)
  - COAP\_MSG\_CONTENT\_FORMAT, [33](#)
  - COAP\_MSG\_CONTENT, [34](#)
  - COAP\_MSG\_CONTINUE, [34](#)
  - COAP\_MSG\_CON, [34](#)
  - COAP\_MSG\_CREATED, [34](#)
  - COAP\_MSG\_DELETED, [34](#)
  - COAP\_MSG\_DELETE, [33](#)
  - COAP\_MSG\_ETAG, [33](#)
  - COAP\_MSG\_FORBIDDEN, [33](#)
  - COAP\_MSG\_GATEWAY\_TIMEOUT, [34](#)
  - COAP\_MSG\_GET, [33](#)
  - COAP\_MSG\_IF\_MATCH, [33](#)
  - COAP\_MSG\_IF\_NONE\_MATCH, [33](#)
  - COAP\_MSG\_INCOMPLETE, [33](#)
  - COAP\_MSG\_INT\_SERVER\_ERR, [34](#)
  - COAP\_MSG\_LOCATION\_PATH, [33](#)
  - COAP\_MSG\_LOCATION\_QUERY, [33](#)
  - COAP\_MSG\_MAX\_AGE, [33](#)
  - COAP\_MSG\_MAX\_BUF\_LEN, [30](#)
  - COAP\_MSG\_MAX\_CODE\_CLASS, [30](#)
  - COAP\_MSG\_MAX\_CODE\_DETAIL, [30](#)
  - COAP\_MSG\_MAX\_MSG\_ID, [30](#)
  - COAP\_MSG\_MAX\_TOKEN\_LEN, [31](#)
  - COAP\_MSG\_METHOD\_NOT\_ALLOWED, [33](#)
  - COAP\_MSG\_NOT\_ACCEPTABLE, [33](#)
  - COAP\_MSG\_NOT\_FOUND, [33](#)

- COAP\_MSG\_NOT\_IMPL, 34
- COAP\_MSG\_NON, 34
- COAP\_MSG\_OP\_MAX\_BLOCK\_SIZE, 31
- COAP\_MSG\_OP\_MAX\_BLOCK\_VAL\_LEN, 31
- COAP\_MSG\_OP\_URI\_PATH\_MAX\_LEN, 32
- COAP\_MSG\_POST, 33
- COAP\_MSG\_PRECOND\_FAILED, 33
- COAP\_MSG\_PROXY\_NOT\_SUP, 34
- COAP\_MSG\_PROXY\_SCHEME, 33
- COAP\_MSG\_PROXY\_URI, 33
- COAP\_MSG\_PUT, 33
- COAP\_MSG\_REQ\_ENT\_TOO\_LARGE, 33
- COAP\_MSG\_REQ, 32
- COAP\_MSG\_RST, 34
- COAP\_MSG\_SERV\_UNAVAIL, 34
- COAP\_MSG\_SERVER\_ERR, 32
- COAP\_MSG\_SIZE1, 33
- COAP\_MSG\_SIZE2, 33
- COAP\_MSG\_SUCCESS, 32
- COAP\_MSG\_UNAUTHORIZED, 33
- COAP\_MSG\_UNSUP\_CONT\_FMT, 33
- COAP\_MSG\_URI\_HOST, 33
- COAP\_MSG\_URI\_PATH, 33
- COAP\_MSG\_URI\_PORT, 33
- COAP\_MSG\_URI\_QUERY, 33
- COAP\_MSG\_VALID, 34
- COAP\_MSG\_VER, 32
- coap\_msg\_add\_op, 34
- coap\_msg\_check\_critical\_ops, 35
- coap\_msg\_check\_unsafe\_ops, 35
- coap\_msg\_class\_t, 32
- coap\_msg\_client\_err\_t, 32
- coap\_msg\_copy, 36
- coap\_msg\_create, 36
- coap\_msg\_destroy, 36
- coap\_msg\_format, 36
- coap\_msg\_gen\_rand\_str, 37
- coap\_msg\_get\_code\_class, 29
- coap\_msg\_get\_code\_detail, 29
- coap\_msg\_get\_first\_op, 29
- coap\_msg\_get\_msg\_id, 29
- coap\_msg\_get\_payload, 29
- coap\_msg\_get\_payload\_len, 30
- coap\_msg\_get\_token, 30
- coap\_msg\_get\_token\_len, 30
- coap\_msg\_get\_type, 30
- coap\_msg\_get\_ver, 30
- coap\_msg\_is\_empty, 30
- coap\_msg\_method\_t, 33
- coap\_msg\_op\_format\_block\_val, 37
- coap\_msg\_op\_get\_len, 31
- coap\_msg\_op\_get\_next, 31
- coap\_msg\_op\_get\_num, 31
- coap\_msg\_op\_get\_val, 31
- coap\_msg\_op\_num\_is\_critical, 31
- coap\_msg\_op\_num\_is\_recognized, 37
- coap\_msg\_op\_num\_is\_unsafe, 31
- coap\_msg\_op\_num\_no\_cache\_key, 31
- coap\_msg\_op\_num\_t, 33
- coap\_msg\_op\_parse\_block\_val, 38
- coap\_msg\_op\_set\_len, 32
- coap\_msg\_op\_set\_next, 32
- coap\_msg\_op\_set\_num, 32
- coap\_msg\_op\_set\_val, 32
- coap\_msg\_parse, 38
- coap\_msg\_parse\_type\_msg\_id, 39
- coap\_msg\_reset, 39
- coap\_msg\_server\_err\_t, 33
- coap\_msg\_set\_code, 39
- coap\_msg\_set\_msg\_id, 40
- coap\_msg\_set\_payload, 40
- coap\_msg\_set\_token, 41
- coap\_msg\_set\_type, 41
- coap\_msg\_success\_t, 34
- coap\_msg\_type\_t, 34
- coap\_msg\_uri\_path\_to\_str, 42
- coap\_msg\_add\_op
  - coap\_msg.c, 54
  - coap\_msg.h, 34
- coap\_msg\_check\_critical\_ops
  - coap\_msg.c, 54
  - coap\_msg.h, 35
- coap\_msg\_check\_unsafe\_ops
  - coap\_msg.c, 55
  - coap\_msg.h, 35
- coap\_msg\_class\_t
  - coap\_msg.h, 32
- coap\_msg\_client\_err\_t
  - coap\_msg.h, 32
- coap\_msg\_copy
  - coap\_msg.c, 55
  - coap\_msg.h, 36
- coap\_msg\_create
  - coap\_msg.c, 55
  - coap\_msg.h, 36
- coap\_msg\_destroy
  - coap\_msg.c, 56
  - coap\_msg.h, 36
- coap\_msg\_format
  - coap\_msg.c, 56
  - coap\_msg.h, 36
- coap\_msg\_gen\_rand\_str
  - coap\_msg.c, 56
  - coap\_msg.h, 37
- coap\_msg\_get\_code\_class
  - coap\_msg.h, 29
- coap\_msg\_get\_code\_detail
  - coap\_msg.h, 29
- coap\_msg\_get\_first\_op
  - coap\_msg.h, 29
- coap\_msg\_get\_msg\_id
  - coap\_msg.h, 29
- coap\_msg\_get\_payload
  - coap\_msg.h, 29
- coap\_msg\_get\_payload\_len
  - coap\_msg.h, 30

- coap\_msg\_get\_token
  - coap\_msg.h, [30](#)
- coap\_msg\_get\_token\_len
  - coap\_msg.h, [30](#)
- coap\_msg\_get\_type
  - coap\_msg.h, [30](#)
- coap\_msg\_get\_ver
  - coap\_msg.h, [30](#)
- coap\_msg\_is\_empty
  - coap\_msg.h, [30](#)
- coap\_msg\_method\_t
  - coap\_msg.h, [33](#)
- coap\_msg\_op, [7](#)
  - len, [7](#)
  - next, [7](#)
  - num, [7](#)
  - val, [7](#)
- coap\_msg\_op\_format\_block\_val
  - coap\_msg.c, [56](#)
  - coap\_msg.h, [37](#)
- coap\_msg\_op\_get\_len
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_get\_next
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_get\_num
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_get\_val
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_list\_get\_first
  - coap\_msg.c, [54](#)
- coap\_msg\_op\_list\_get\_last
  - coap\_msg.c, [54](#)
- coap\_msg\_op\_list\_is\_empty
  - coap\_msg.c, [54](#)
- coap\_msg\_op\_list\_t, [8](#)
  - first, [8](#)
  - last, [8](#)
- coap\_msg\_op\_num\_is\_critical
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_num\_is\_recognized
  - coap\_msg.c, [57](#)
  - coap\_msg.h, [37](#)
- coap\_msg\_op\_num\_is\_unsafe
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_num\_no\_cache\_key
  - coap\_msg.h, [31](#)
- coap\_msg\_op\_num\_t
  - coap\_msg.h, [33](#)
- coap\_msg\_op\_parse\_block\_val
  - coap\_msg.c, [57](#)
  - coap\_msg.h, [38](#)
- coap\_msg\_op\_set\_len
  - coap\_msg.h, [32](#)
- coap\_msg\_op\_set\_next
  - coap\_msg.h, [32](#)
- coap\_msg\_op\_set\_num
  - coap\_msg.h, [32](#)
- coap\_msg\_op\_set\_val
  - coap\_msg.h, [32](#)
- coap\_msg\_parse
  - coap\_msg.c, [58](#)
  - coap\_msg.h, [38](#)
- coap\_msg\_parse\_type\_msg\_id
  - coap\_msg.c, [58](#)
  - coap\_msg.h, [39](#)
- coap\_msg\_reset
  - coap\_msg.c, [59](#)
  - coap\_msg.h, [39](#)
- coap\_msg\_server\_err\_t
  - coap\_msg.h, [33](#)
- coap\_msg\_set\_code
  - coap\_msg.c, [59](#)
  - coap\_msg.h, [39](#)
- coap\_msg\_set\_msg\_id
  - coap\_msg.c, [59](#)
  - coap\_msg.h, [40](#)
- coap\_msg\_set\_payload
  - coap\_msg.c, [60](#)
  - coap\_msg.h, [40](#)
- coap\_msg\_set\_token
  - coap\_msg.c, [60](#)
  - coap\_msg.h, [41](#)
- coap\_msg\_set\_type
  - coap\_msg.c, [61](#)
  - coap\_msg.h, [41](#)
- coap\_msg\_success\_t
  - coap\_msg.h, [34](#)
- coap\_msg\_t, [9](#)
  - code\_class, [10](#)
  - code\_detail, [10](#)
  - msg\_id, [10](#)
  - op\_list, [10](#)
  - payload, [10](#)
  - payload\_len, [10](#)
  - token, [10](#)
  - token\_len, [10](#)
  - type, [10](#)
  - ver, [10](#)
- coap\_msg\_type\_t
  - coap\_msg.h, [34](#)
- coap\_msg\_uri\_path\_to\_str
  - coap\_msg.c, [61](#)
  - coap\_msg.h, [42](#)
- coap\_server, [11](#)
  - cred, [12](#)
  - dh\_params, [12](#)
  - handle, [12](#)
  - msg\_id, [12](#)
  - priority, [12](#)
  - sd, [12](#)
  - sep\_list, [12](#)
  - trans, [12](#)
- coap\_server.c
  - COAP\_SERVER\_ACK\_TIMEOUT\_SEC, [63](#)
  - COAP\_SERVER\_DTLS\_MTU, [63](#)
  - COAP\_SERVER\_DTLS\_NUM\_DH\_BITS, [63](#)

- COAP\_SERVER\_DTLS\_PRIORITIES, 63
- COAP\_SERVER\_DTLS\_RETRANS\_TIMEOUT, 63
- COAP\_SERVER\_DTLS\_TOTAL\_TIMEOUT, 63
- COAP\_SERVER\_MAX\_RETRANSMIT, 63
- coap\_server\_add\_sep\_resp\_uri\_path, 63
- coap\_server\_create, 64
- coap\_server\_destroy, 64
- coap\_server\_get\_next\_msg\_id, 65
- coap\_server\_run, 65
- coap\_server.h
  - COAP\_SERVER\_ADDR\_BUF\_LEN, 44
  - COAP\_SERVER\_DIAG\_PAYLOAD\_LEN, 44
  - COAP\_SERVER\_NUM\_TRANS, 44
  - COAP\_SERVER\_PIGGYBACKED, 44
  - COAP\_SERVER\_SEPARATE, 44
  - coap\_server\_add\_sep\_resp\_uri\_path, 44
  - coap\_server\_create, 45
  - coap\_server\_destroy, 45
  - coap\_server\_get\_next\_msg\_id, 46
  - coap\_server\_resp\_t, 44
  - coap\_server\_run, 46
- coap\_server\_add\_sep\_resp\_uri\_path
  - coap\_server.c, 63
  - coap\_server.h, 44
- coap\_server\_create
  - coap\_server.c, 64
  - coap\_server.h, 45
- coap\_server\_destroy
  - coap\_server.c, 64
  - coap\_server.h, 45
- coap\_server\_get\_next\_msg\_id
  - coap\_server.c, 65
  - coap\_server.h, 46
- coap\_server\_path, 13
  - next, 13
  - str, 13
- coap\_server\_path\_list\_t, 14
  - first, 14
  - last, 14
- coap\_server\_resp\_t
  - coap\_server.h, 44
- coap\_server\_run
  - coap\_server.c, 65
  - coap\_server.h, 46
- coap\_server\_trans, 15
  - active, 16
  - client\_addr, 16
  - client\_sin, 16
  - client\_sin\_len, 16
  - last\_use, 16
  - num\_retrans, 16
  - req, 16
  - resp, 16
  - server, 16
  - session, 16
  - timeout, 17
  - timer\_fd, 17
- code\_class
  - coap\_msg\_t, 10
- code\_detail
  - coap\_msg\_t, 10
- cred
  - coap\_client\_t, 5
  - coap\_server, 12
- dh\_params
  - coap\_server, 12
- first
  - coap\_msg\_op\_list\_t, 8
  - coap\_server\_path\_list\_t, 14
- handle
  - coap\_server, 12
- last
  - coap\_msg\_op\_list\_t, 8
  - coap\_server\_path\_list\_t, 14
- last\_use
  - coap\_server\_trans, 16
- len
  - coap\_msg\_op, 7
- lib/include/coap\_client.h, 19
- lib/include/coap\_ipv.h, 21
- lib/include/coap\_log.h, 23
- lib/include/coap\_msg.h, 26
- lib/include/coap\_server.h, 42
- lib/src/coap\_client.c, 46
- lib/src/coap\_log.c, 49
- lib/src/coap\_msg.c, 52
- lib/src/coap\_server.c, 61
- msg\_id
  - coap\_msg\_t, 10
  - coap\_server, 12
- next
  - coap\_msg\_op, 7
  - coap\_server\_path, 13
- num
  - coap\_msg\_op, 7
- num\_retrans
  - coap\_client\_t, 5
  - coap\_server\_trans, 16
- op\_list
  - coap\_msg\_t, 10
- payload
  - coap\_msg\_t, 10
- payload\_len
  - coap\_msg\_t, 10
- priority
  - coap\_client\_t, 6
  - coap\_server, 12
- req

- coap\_server\_trans, [16](#)
- resp
  - coap\_server\_trans, [16](#)
- sd
  - coap\_client\_t, [6](#)
  - coap\_server, [12](#)
- sep\_list
  - coap\_server, [12](#)
- server
  - coap\_server\_trans, [16](#)
- server\_host
  - coap\_client\_t, [6](#)
- server\_port
  - coap\_client\_t, [6](#)
- server\_sin
  - coap\_client\_t, [6](#)
- server\_sin\_len
  - coap\_client\_t, [6](#)
- session
  - coap\_client\_t, [6](#)
  - coap\_server\_trans, [16](#)
- str
  - coap\_server\_path, [13](#)
- timeout
  - coap\_client\_t, [6](#)
  - coap\_server\_trans, [17](#)
- timer\_fd
  - coap\_client\_t, [6](#)
  - coap\_server\_trans, [17](#)
- token
  - coap\_msg\_t, [10](#)
- token\_len
  - coap\_msg\_t, [10](#)
- trans
  - coap\_server, [12](#)
- type
  - coap\_msg\_t, [10](#)
- val
  - coap\_msg\_op, [7](#)
- ver
  - coap\_msg\_t, [10](#)