

FreeCoAP

Generated by Doxygen 1.8.10

Sat Aug 15 2015 14:40:54



# Contents

<b>1</b>	<b>README</b>	<b>1</b>
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	coap_client_t Struct Reference . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.2	coap_msg_op Struct Reference . . . . .	7
4.2.1	Detailed Description . . . . .	8
4.3	coap_msg_op_list_t Struct Reference . . . . .	8
4.3.1	Detailed Description . . . . .	8
4.4	coap_msg_t Struct Reference . . . . .	8
4.4.1	Detailed Description . . . . .	8
4.5	coap_server Struct Reference . . . . .	9
4.5.1	Detailed Description . . . . .	9
4.6	coap_server_path Struct Reference . . . . .	9
4.6.1	Detailed Description . . . . .	9
4.7	coap_server_path_list_t Struct Reference . . . . .	9
4.7.1	Detailed Description . . . . .	10
4.8	coap_server_trans Struct Reference . . . . .	10
4.8.1	Detailed Description . . . . .	10
4.9	harness_test Struct Reference . . . . .	10
4.10	test_data Struct Reference . . . . .	10
4.11	test_op Struct Reference . . . . .	11
<b>5</b>	<b>File Documentation</b>	<b>13</b>
5.1	include/coap_client.h File Reference . . . . .	13
5.1.1	Detailed Description . . . . .	13
5.1.2	Macro Definition Documentation . . . . .	13

5.1.2.1	COAP_CLIENT_ADDR_BUF_LEN	13
5.1.3	Function Documentation	14
5.1.3.1	coap_client_create(coap_client_t *client, const char *host, unsigned port)	14
5.1.3.2	coap_client_destroy(coap_client_t *client)	14
5.1.3.3	coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)	14
5.2	include/coap_log.h File Reference	15
5.2.1	Detailed Description	15
5.2.2	Macro Definition Documentation	15
5.2.2.1	COAP_LOG_DEF_LEVEL	15
5.2.3	Enumeration Type Documentation	16
5.2.3.1	coap_log_level_t	16
5.2.4	Function Documentation	16
5.2.4.1	coap_log_debug(const char *msg,...)	16
5.2.4.2	coap_log_error(const char *msg,...)	16
5.2.4.3	coap_log_info(const char *msg,...)	16
5.2.4.4	coap_log_notice(const char *msg,...)	16
5.2.4.5	coap_log_set_level(coap_log_level_t level)	16
5.2.4.6	coap_log_warn(const char *msg,...)	17
5.3	include/coap_msg.h File Reference	17
5.3.1	Detailed Description	19
5.3.2	Macro Definition Documentation	19
5.3.2.1	coap_msg_get_code_class	19
5.3.2.2	coap_msg_get_code_detail	19
5.3.2.3	coap_msg_get_first_op	19
5.3.2.4	coap_msg_get_msg_id	19
5.3.2.5	coap_msg_get_payload	19
5.3.2.6	coap_msg_get_payload_len	19
5.3.2.7	coap_msg_get_token	20
5.3.2.8	coap_msg_get_token_len	20
5.3.2.9	coap_msg_get_type	20
5.3.2.10	coap_msg_get_ver	20
5.3.2.11	coap_msg_is_empty	20
5.3.2.12	COAP_MSG_MAX_BUF_LEN	20
5.3.2.13	COAP_MSG_MAX_CODE_CLASS	20
5.3.2.14	COAP_MSG_MAX_CODE_DETAIL	20
5.3.2.15	COAP_MSG_MAX_MSG_ID	20
5.3.2.16	COAP_MSG_MAX_TOKEN_LEN	20
5.3.2.17	coap_msg_op_get_len	20
5.3.2.18	coap_msg_op_get_next	20
5.3.2.19	coap_msg_op_get_num	21

5.3.2.20	<code>coap_msg_op_get_val</code> . . . . .	21
5.3.2.21	<code>coap_msg_op_num_is_critical</code> . . . . .	21
5.3.2.22	<code>coap_msg_op_num_is_unsafe</code> . . . . .	21
5.3.2.23	<code>coap_msg_op_num_no_cache_key</code> . . . . .	21
5.3.2.24	<code>coap_msg_op_set_len</code> . . . . .	21
5.3.2.25	<code>coap_msg_op_set_next</code> . . . . .	21
5.3.2.26	<code>coap_msg_op_set_num</code> . . . . .	21
5.3.2.27	<code>coap_msg_op_set_val</code> . . . . .	21
5.3.2.28	<code>COAP_MSG_OP_URI_PATH_MAX_LEN</code> . . . . .	21
5.3.2.29	<code>COAP_MSG_OP_URI_PATH_NUM</code> . . . . .	21
5.3.2.30	<code>COAP_MSG_VER</code> . . . . .	21
5.3.3	Function Documentation . . . . .	22
5.3.3.1	<code>coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, char *val)</code> . . . . .	22
5.3.3.2	<code>coap_msg_copy(coap_msg_t *dst, coap_msg_t *src)</code> . . . . .	23
5.3.3.3	<code>coap_msg_create(coap_msg_t *msg)</code> . . . . .	23
5.3.3.4	<code>coap_msg_destroy(coap_msg_t *msg)</code> . . . . .	23
5.3.3.5	<code>coap_msg_format(coap_msg_t *msg, char *buf, unsigned len)</code> . . . . .	23
5.3.3.6	<code>coap_msg_gen_rand_str(char *buf, unsigned len)</code> . . . . .	24
5.3.3.7	<code>coap_msg_parse(coap_msg_t *msg, char *buf, unsigned len)</code> . . . . .	24
5.3.3.8	<code>coap_msg_parse_type_msg_id(char *buf, unsigned len, unsigned *type, unsigned *msg_id)</code> . . . . .	24
5.3.3.9	<code>coap_msg_reset(coap_msg_t *msg)</code> . . . . .	25
5.3.3.10	<code>coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code_detail)</code> . . . . .	25
5.3.3.11	<code>coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id)</code> . . . . .	25
5.3.3.12	<code>coap_msg_set_payload(coap_msg_t *msg, char *buf, unsigned len)</code> . . . . .	26
5.3.3.13	<code>coap_msg_set_token(coap_msg_t *msg, char *buf, unsigned len)</code> . . . . .	26
5.3.3.14	<code>coap_msg_set_type(coap_msg_t *msg, unsigned type)</code> . . . . .	26
5.4	<code>include/coap_server.h</code> File Reference . . . . .	27
5.4.1	Detailed Description . . . . .	28
5.4.2	Macro Definition Documentation . . . . .	28
5.4.2.1	<code>COAP_SERVER_ADDR_BUF_LEN</code> . . . . .	28
5.4.2.2	<code>COAP_SERVER_NUM_TRANS</code> . . . . .	28
5.4.3	Function Documentation . . . . .	28
5.4.3.1	<code>coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str)</code> . . . . .	28
5.4.3.2	<code>coap_server_create(coap_server_t *server, int(*handle)(coap_server_t *, coap_msg_t *, coap_msg_t *), const char *host, unsigned port)</code> . . . . .	28
5.4.3.3	<code>coap_server_destroy(coap_server_t *server)</code> . . . . .	29
5.4.3.4	<code>coap_server_get_next_msg_id(coap_server_t *server)</code> . . . . .	29
5.4.3.5	<code>coap_server_run(coap_server_t *server)</code> . . . . .	29
5.5	<code>src/coap_client.c</code> File Reference . . . . .	29

5.5.1	Detailed Description	30
5.5.2	Macro Definition Documentation	30
5.5.2.1	COAP_CLIENT_ACK_TIMEOUT_SEC	30
5.5.2.2	COAP_CLIENT_MAX_RETRANSMIT	30
5.5.2.3	COAP_CLIENT_RESP_TIMEOUT_SEC	30
5.5.3	Function Documentation	30
5.5.3.1	coap_client_create(coap_client_t *client, const char *host, unsigned port)	30
5.5.3.2	coap_client_destroy(coap_client_t *client)	31
5.5.3.3	coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)	31
5.6	src/coap_msg.c File Reference	31
5.6.1	Detailed Description	32
5.6.2	Macro Definition Documentation	32
5.6.2.1	coap_msg_op_list_get_first	32
5.6.2.2	coap_msg_op_list_get_last	32
5.6.2.3	coap_msg_op_list_is_empty	33
5.6.3	Function Documentation	33
5.6.3.1	coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, char *val)	33
5.6.3.2	coap_msg_copy(coap_msg_t *dst, coap_msg_t *src)	33
5.6.3.3	coap_msg_create(coap_msg_t *msg)	33
5.6.3.4	coap_msg_destroy(coap_msg_t *msg)	33
5.6.3.5	coap_msg_format(coap_msg_t *msg, char *buf, unsigned len)	34
5.6.3.6	coap_msg_gen_rand_str(char *buf, unsigned len)	34
5.6.3.7	coap_msg_parse(coap_msg_t *msg, char *buf, unsigned len)	34
5.6.3.8	coap_msg_parse_type_msg_id(char *buf, unsigned len, unsigned *type, unsigned *msg_id)	35
5.6.3.9	coap_msg_reset(coap_msg_t *msg)	35
5.6.3.10	coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code←_detail)	35
5.6.3.11	coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id)	35
5.6.3.12	coap_msg_set_payload(coap_msg_t *msg, char *buf, unsigned len)	36
5.6.3.13	coap_msg_set_token(coap_msg_t *msg, char *buf, unsigned len)	36
5.6.3.14	coap_msg_set_type(coap_msg_t *msg, unsigned type)	36
5.7	src/coap_server.c File Reference	37
5.7.1	Detailed Description	38
5.7.2	Macro Definition Documentation	38
5.7.2.1	COAP_SERVER_ACK_TIMEOUT_SEC	38
5.7.2.2	COAP_SERVER_MAX_RETRANSMIT	38
5.7.3	Function Documentation	38
5.7.3.1	coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str)	38
5.7.3.2	coap_server_create(coap_server_t *server, int(*handle)(coap_server_t *, coap_msg_t *, coap_msg_t *), const char *host, unsigned port)	38

---

5.7.3.3	<code>coap_server_destroy(coap_server_t *server)</code> . . . . .	39
5.7.3.4	<code>coap_server_get_next_msg_id(coap_server_t *server)</code> . . . . .	39
5.7.3.5	<code>coap_server_run(coap_server_t *server)</code> . . . . .	39
<b>Index</b>		<b>41</b>





# Chapter 1

## README

=====

### FreeCoAP

An implementation of the CoAP protocol for GNU/Linux consisting of:

- a CoAP message parser/formatter library
- a CoAP client library
- a CoAP server library
- unit test code for the message parser/formatter
- a client test application
- a server test application

Copyright (c) 2015 Keith Cullen

Released under a BSD style license.

### Test Instructions

#### To build and test the message parser/formatter

```
$ cd coap/test_msg
```

```
$ make
```

```
$ ./test_msg
```

#### To build and run the server test application

```
$ cd coap/test_server
```

```
$ make
```

```
$ ./server
```

**To build and run the client test application**

```
$ cd coap/test_client  
$ make  
$ ./client
```

**To build and run the server test application with DTLS enabled**

```
$ cd coap/test_server  
$ make dtls=y  
$ ./server
```

**To build and run the client test application with DTLS enabled**

```
$ cd coap/test_client  
$ make dtls=y  
$ ./client
```

## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">coap_client_t</a>		
Client structure	. . . . .	7
<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	. . . . .	8
<a href="#">coap_server</a>		
Server structure	. . . . .	9
<a href="#">coap_server_path</a>		
URI path structure	. . . . .	9
<a href="#">coap_server_path_list_t</a>		
URI path list structure	. . . . .	9
<a href="#">coap_server_trans</a>		
Transaction structure	. . . . .	10
<a href="#">harness_test</a>	. . . . .	10
<a href="#">test_data</a>	. . . . .	10
<a href="#">test_op</a>	. . . . .	11



## Chapter 3

# File Index

### 3.1 File List

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

include/ <a href="#">coap_client.h</a>	
Include file for the FreeCoAP client library . . . . .	13
include/ <a href="#">coap_log.h</a>	
Include file for the FreeCoAP logging module . . . . .	15
include/ <a href="#">coap_msg.h</a>	
Include file for the FreeCoAP message parser/formatter library . . . . .	17
include/ <a href="#">coap_server.h</a>	
Include file for the FreeCoAP server library . . . . .	27
src/ <a href="#">coap_client.c</a>	
Source file for the FreeCoAP client library . . . . .	29
src/ <a href="#">coap_msg.c</a>	
Source file for the FreeCoAP message parser/formatter library . . . . .	31
src/ <a href="#">coap_server.c</a>	
Source file for the FreeCoAP server library . . . . .	37
test_msg/ <a href="#">harness.h</a> . . . . .	??



## Chapter 4

# Class Documentation

### 4.1 coap\_client\_t Struct Reference

Client structure.

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

#### Public Attributes

- int **sd**
- int **timer\_fd**
- struct timespec **timeout**
- unsigned **num\_retrans**
- struct sockaddr\_in6 **server\_sin**
- socklen\_t **server\_sin\_len**
- char **server\_addr** [[COAP\\_CLIENT\\_ADDR\\_BUF\\_LEN](#)]

#### 4.1.1 Detailed Description

Client structure.

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

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

### 4.2 coap\_msg\_op Struct Reference

Option structure.

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

#### Public Attributes

- unsigned **num**
- unsigned **len**
- char \* **val**
- struct [coap\\_msg\\_op](#) \* **next**

### 4.2.1 Detailed Description

Option structure.

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

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

## 4.3 coap\_msg\_op\_list\_t Struct Reference

Option linked-list structure.

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

### Public Attributes

- [coap\\_msg\\_op\\_t](#) \* **first**
- [coap\\_msg\\_op\\_t](#) \* **last**

### 4.3.1 Detailed Description

Option linked-list structure.

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

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

## 4.4 coap\_msg\_t Struct Reference

Message structure.

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

### 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**
- unsigned **payload\_len**

### 4.4.1 Detailed Description

Message structure.

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

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



## 4.5 coap\_server Struct Reference

Server structure.

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

### 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](#) \*)

### 4.5.1 Detailed Description

Server structure.

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

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

## 4.6 coap\_server\_path Struct Reference

URI path structure.

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

### Public Attributes

- char \* **str**
- struct [coap\\_server\\_path](#) \* **next**

### 4.6.1 Detailed Description

URI path structure.

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

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

## 4.7 coap\_server\_path\_list\_t Struct Reference

URI path list structure.

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

### Public Attributes

- [coap\\_server\\_path\\_t](#) \* **first**
- [coap\\_server\\_path\\_t](#) \* **last**

### 4.7.1 Detailed Description

URI path list structure.

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

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

## 4.8 coap\_server\_trans Struct Reference

Transaction structure.

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

### Public Attributes

- int **active**
- time\_t **last\_use**
- int **timer\_fd**
- struct timespec **timeout**
- unsigned **num\_retrans**
- struct sockaddr\_in6 **client\_sin**
- socklen\_t **client\_sin\_len**
- char **client\_addr** [[COAP\\_SERVER\\_ADDR\\_BUF\\_LEN](#)]
- [coap\\_msg\\_t](#) **req**
- [coap\\_msg\\_t](#) **resp**
- struct [coap\\_server](#) \* **server**

### 4.8.1 Detailed Description

Transaction structure.

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

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

## 4.9 harness\_test Struct Reference

### Public Attributes

- harness\_func **func**
- harness\_data **data**

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

- [test\\_msg/harness.h](#)

## 4.10 test\_data Struct Reference

### Public Attributes

- const char \* **parse\_desc**

- const char \* **format\_desc**
- const char \* **copy\_desc**
- int **parse\_ret**
- int **set\_type\_ret**
- int **set\_code\_ret**
- int **set\_msg\_id\_ret**
- int **set\_token\_ret**
- int \* **add\_op\_ret**
- int **set\_payload\_ret**
- int **format\_ret**
- int **copy\_ret**
- char \* **buf**
- unsigned **buf\_len**
- unsigned **ver**
- [coap\\_msg\\_type\\_t](#) type
- unsigned **code\_class**
- unsigned **code\_detail**
- unsigned **msg\_id**
- char \* **token**
- unsigned **token\_len**
- struct [test\\_op](#) \* **ops**
- unsigned **num\_ops**
- char \* **payload**
- unsigned **payload\_len**

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

- test\_msg/test\_msg.c

## 4.11 test\_op Struct Reference

### Public Attributes

- unsigned **num**
- unsigned **len**
- char \* **val**

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

- test\_msg/test\_msg.c



## Chapter 5

# File Documentation

### 5.1 include/coap\_client.h File Reference

Include file for the FreeCoAP client library.

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

#### Classes

- struct [coap\\_client\\_t](#)  
*Client structure.*

#### Macros

- #define [COAP\\_CLIENT\\_ADDR\\_BUF\\_LEN](#) 128

#### Functions

- int [coap\\_client\\_create](#) ([coap\\_client\\_t](#) \*client, const char \*host, unsigned port)  
*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.*

#### 5.1.1 Detailed Description

Include file for the FreeCoAP client library.

#### 5.1.2 Macro Definition Documentation

##### 5.1.2.1 #define COAP\_CLIENT\_ADDR\_BUF\_LEN 128

Buffer length for host addresses

### 5.1.3 Function Documentation

#### 5.1.3.1 `int coap_client_create ( coap_client_t * client, const char * host, unsigned port )`

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

##### Returns

Operation status

##### Return values

0	Success
-errno	On error

#### 5.1.3.2 `void coap_client_destroy ( coap_client_t * client )`

Deinitialise a client structure.

##### Parameters

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

#### 5.1.3.3 `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.

##### Parameters

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

##### Returns

Operation status

##### Return values

0	Success
-errno	On Error

##### 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

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

##### Returns

Operation status

## Return values

0	Success
-errno	Error

## 5.2 include/coap\_log.h File Reference

Include file for the FreeCoAP logging module.

### 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.*
- 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.*

#### 5.2.1 Detailed Description

Include file for the FreeCoAP logging module.

Source file for the FreeCoAP logging module.

#### 5.2.2 Macro Definition Documentation

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

Default log level

### 5.2.3 Enumeration Type Documentation

#### 5.2.3.1 enum coap\_log\_level\_t

Log level.

Enumerator

**COAP\_LOG\_ERROR** Highest severity level

**COAP\_LOG\_DEBUG** Lowest severity level

### 5.2.4 Function Documentation

#### 5.2.4.1 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

#### 5.2.4.2 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

#### 5.2.4.3 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

#### 5.2.4.4 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

#### 5.2.4.5 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
----	--------------	-------------------

5.2.4.6 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

## 5.3 include/coap\_msg.h File Reference

Include file for the FreeCoAP message parser/formatter library.

### Classes

- struct [coap\\_msg\\_op](#)  
*Option structure.*
- struct [coap\\_msg\\_op\\_list\\_t](#)  
*Option linked-list structure.*
- struct [coap\\_msg\\_t](#)  
*Message structure.*

### Macros

- #define [COAP\\_MSG\\_VER](#) 0x01
- #define [COAP\\_MSG\\_MAX\\_TOKEN\\_LEN](#) 8
- #define [COAP\\_MSG\\_MAX\\_CODE\\_CLASS](#) 7
- #define [COAP\\_MSG\\_MAX\\_CODE\\_DETAIL](#) 31
- #define [COAP\\_MSG\\_MAX\\_MSG\\_ID](#) ((1 << 16) - 1)
- #define [COAP\\_MSG\\_OP\\_URI\\_PATH\\_NUM](#) 11
- #define [COAP\\_MSG\\_OP\\_URI\\_PATH\\_MAX\\_LEN](#) 256
- #define [COAP\\_MSG\\_MAX\\_BUF\\_LEN](#) 1152
- #define [coap\\_msg\\_op\\_num\\_is\\_critical](#)(num) ((num) & 1)
- #define [coap\\_msg\\_op\\_num\\_is\\_unsafe](#)(num) ((num) & 2)
- #define [coap\\_msg\\_op\\_num\\_no\\_cache\\_key](#)(num) ((num & 0x1e) == 0x1c)
- #define [coap\\_msg\\_op\\_get\\_num](#)(op) ((op)->num)
- #define [coap\\_msg\\_op\\_set\\_num](#)(op, num) ((op)->num = (num))
- #define [coap\\_msg\\_op\\_get\\_len](#)(op) ((op)->len)
- #define [coap\\_msg\\_op\\_set\\_len](#)(op, len) ((op)->len = (len))
- #define [coap\\_msg\\_op\\_get\\_val](#)(op) ((op)->val)
- #define [coap\\_msg\\_op\\_set\\_val](#)(op, val) ((op)->val = (val))
- #define [coap\\_msg\\_op\\_get\\_next](#)(op) ((op)->next)
- #define [coap\\_msg\\_op\\_set\\_next](#)(op, next\_op) ((op)->next = (next\_op))
- #define [coap\\_msg\\_get\\_ver](#)(msg) ((msg)->ver)
- #define [coap\\_msg\\_get\\_type](#)(msg) ((msg)->type)
- #define [coap\\_msg\\_get\\_token\\_len](#)(msg) ((msg)->token\_len)
- #define [coap\\_msg\\_get\\_code\\_class](#)(msg) ((msg)->code\_class)
- #define [coap\\_msg\\_get\\_code\\_detail](#)(msg) ((msg)->code\_detail)

- #define `coap_msg_get_msg_id(msg)` `((msg)->msg_id)`
- #define `coap_msg_get_token(msg)` `((msg)->token)`
- #define `coap_msg_get_first_op(msg)` `((msg)->op_list.first)`
- #define `coap_msg_get_payload(msg)` `((msg)->payload)`
- #define `coap_msg_get_payload_len(msg)` `((msg)->payload_len)`
- #define `coap_msg_is_empty(msg)` `((msg)->code_class == 0) && ((msg)->code_detail == 0)`

## Typedefs

- typedef struct `coap_msg_op` `coap_msg_op_t`  
*Option structure.*

## Enumerations

- enum `coap_msg_type_t` { `COAP_MSG_CON` = 0x0, `COAP_MSG_NON` = 0x1, `COAP_MSG_ACK` = 0x2, `COAP_MSG_RST` = 0x3 }  
*Message type enumeration.*
- enum `coap_msg_class_t` { `COAP_MSG_REQ` = 0, `COAP_MSG_SUCCESS` = 2, `COAP_MSG_CLIENT_ERR` = 4, `COAP_MSG_SERVER_ERR` = 5 }  
*Code class enumeration.*
- enum `coap_msg_method_t` { `COAP_MSG_GET` = 1, `COAP_MSG_POST` = 2, `COAP_MSG_PUT` = 3, `COAP_MSG_DELETE` = 4 }  
*Code detail enumeration.*
- enum `coap_msg_success_t` { `COAP_MSG_CREATED` = 1, `COAP_MSG_DELETED` = 2, `COAP_MSG_VALID` = 3, `COAP_MSG_CHANGED` = 4, `COAP_MSG_CONTENT` = 5 }  
*Success response code detail enumeration.*
- enum `coap_msg_client_err_t` { `COAP_MSG_BAD_REQ` = 0, `COAP_MSG_UNAUTHORIZED` = 1, `COAP_MSG_BAD_OPTION` = 2, `COAP_MSG_FORBIDDEN` = 3, `COAP_MSG_NOT_FOUND` = 4, `COAP_MSG_METHOD_NOT_ALLOWED` = 5, `COAP_MSG_NOT_ACCEPTABLE` = 6, `COAP_MSG_PRECOND_FAILED` = 12, `COAP_MSG_REQ_ENT_TOO_LARGE` = 13, `COAP_MSG_UNSUP_CONT_FMT` = 15 }  
*Client error response code detail enumeration.*
- enum `coap_msg_server_err_t` { `COAP_MSG_INT_SERVER_ERR` = 0, `COAP_MSG_NOT_IMPL` = 1, `COAP_MSG_BAD_GATEWAY` = 2, `COAP_MSG_SERV_UNAVAIL` = 3, `COAP_MSG_GATEWAY_TIMEOUT` = 4, `COAP_MSG_PROXY_NOT_SUP` = 5 }  
*Server error response code detail enumeration.*

## Functions

- void `coap_msg_gen_rand_str` (char \*buf, unsigned 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.*
- int `coap_msg_parse_type_msg_id` (char \*buf, unsigned len, unsigned \*type, unsigned \*msg\_id)

- Extract the type and message ID values from a message.*
- int `coap_msg_parse` (`coap_msg_t` \*msg, char \*buf, unsigned 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, unsigned len)  
*Set the token in a message.*
- int `coap_msg_add_op` (`coap_msg_t` \*msg, unsigned num, unsigned len, char \*val)  
*Add a token to a message structure.*
- int `coap_msg_set_payload` (`coap_msg_t` \*msg, char \*buf, unsigned len)  
*Set the payload in a message.*
- int `coap_msg_format` (`coap_msg_t` \*msg, char \*buf, unsigned len)  
*Format a message.*
- int `coap_msg_copy` (`coap_msg_t` \*dst, `coap_msg_t` \*src)  
*Copy a message.*

### 5.3.1 Detailed Description

Include file for the FreeCoAP message parser/formatter library.

### 5.3.2 Macro Definition Documentation

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

Get the code class from a message

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

Get the code detail from a message

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

Get the first option from a message

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

Get the message ID from message

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

Get the payload from a message

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

Get the payload length from a message

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

Get the token from a message

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

Get the token length from a message

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

Get the type from a message

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

Get the version from a message

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

Indicate if a message is empty

5.3.2.12 `#define COAP_MSG_MAX_BUF_LEN 1152`

Maximum buffer length for header and payload

5.3.2.13 `#define COAP_MSG_MAX_CODE_CLASS 7`

Maximum code class

5.3.2.14 `#define COAP_MSG_MAX_CODE_DETAIL 31`

Maximum code detail

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

Maximum message ID

5.3.2.16 `#define COAP_MSG_MAX_TOKEN_LEN 8`

Maximum token length

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

Get the option length from an option

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

Get the next pointer from an option

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

Get the option number from an option

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

Get the option value from an option

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

Indicate if an option is critical

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

Indicate if an option is unsafe to forward

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

Indicate if an option is not part of the cache key

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

Set the option length in an option

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

Set the next pointer in an option

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

Set the option number in an option

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

Set the option value in an option

5.3.2.28 `#define COAP_MSG_OP_URI_PATH_MAX_LEN 256`

Maximum buffer length for a reconstructed URI path

5.3.2.29 `#define COAP_MSG_OP_URI_PATH_NUM 11`

Uri-path option number

5.3.2.30 `#define COAP_MSG_VER 0x01`

CoAP version

### 5.3.3 Function Documentation

5.3.3.1 `int coap_msg_add_op ( coap_msg_t * msg, unsigned num, unsigned len, char * val )`

Add a token to a message structure.

## Parameters

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
-ENOMEM	Out-of-memory

## 5.3.3.2 int coap\_msg\_copy ( coap\_msg\_t \* dst, coap\_msg\_t \* src )

Copy a message.

## Parameters

out	<i>dst</i>	Pointer to the destination message structure
in	<i>src</i>	Pointer to the source message structure

## Returns

Operation status

## Return values

0	Success
-EINVAL	Invalid argument
-ENOMEM	Out-of-memory

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

Initialise a message structure.

## Parameters

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

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

Deinitialise a message structure.

## Parameters

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

## 5.3.3.5 int coap\_msg\_format ( coap\_msg\_t \* msg, char \* buf, unsigned 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**

Operation status

**Return values**

> 0	Length of the formatted message
-ENOSPC	Insufficient buffer length
-EBADMSG	Message is corrupt

**5.3.3.6 void coap\_msg\_gen\_rand\_str ( char \* buf, unsigned 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

**5.3.3.7 int coap\_msg\_parse ( coap\_msg\_t \* msg, char \* buf, unsigned len )**

Parse a message.

**Parameters**

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
-EINVAL	Invalid argument
-ENOMEM	Out-of-memory
-EBADMSG	The message is corrupt

**5.3.3.8 int coap\_msg\_parse\_type\_msg\_id ( char \* buf, unsigned 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
-EBADMSG	The message is corrupt

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

Deinitialise and initialise a message structure.

## Parameters

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

## 5.3.3.10 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
-EINVAL	Invalid argument

## 5.3.3.11 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_id</i>	Message ID

## Returns

Operation status

## Return values

0	Success
-EINVAL	Invalid argument

## 5.3.3.12 int coap\_msg\_set\_payload ( coap\_msg\_t \* msg, char \* buf, unsigned 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

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

## Returns

Operation status

## Return values

0	Success
-ENOMEM	Out-of-memory

## 5.3.3.13 int coap\_msg\_set\_token ( coap\_msg\_t \* msg, char \* buf, unsigned len )

Set the token in a message.

## Parameters

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

## Returns

Operation status

## Return values

0	Success
-EINVAL	Invalid argument

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

Set the type in a message.

## Parameters

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

in	type	Message type
----	------	--------------

**Returns**

Operation status

**Return values**

0	Success
-EINVAL	Invalid argument

## 5.4 include/coap\_server.h File Reference

Include file for the FreeCoAP server library.

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

**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

**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, unsigned port)  
*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.*

### 5.4.1 Detailed Description

Include file for the FreeCoAP server library.

### 5.4.2 Macro Definition Documentation

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

Buffer length for host addresses

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

Maximum number of active transactions per server

### 5.4.3 Function Documentation

#### 5.4.3.1 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	<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
-ENOMEM	Out-of-memory

#### 5.4.3.2 int coap\_server\_create ( coap\_server\_t \* server, int(\*) (coap\_server\_t \*, coap\_msg\_t \*, coap\_msg\_t \*) handle, const char \* host, unsigned port )

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

**Returns**

Operation status

**Return values**

0	Success
-errno	Error

**5.4.3.3 void coap\_server\_destroy ( coap\_server\_t \* server )**

Deinitialise a server structure.

**Parameters**

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

**5.4.3.4 unsigned coap\_server\_get\_next\_msg\_id ( coap\_server\_t \* server )**

Get a new message ID value.

**Parameters**

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

**Returns**

message ID value

**5.4.3.5 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.

**Returns**

Operation status

**Return values**

0	Success
-errno	Error

**5.5 src/coap\_client.c File Reference**

Source file for the FreeCoAP client library.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <sys/select.h>
#include <sys/timerfd.h>
#include "coap_client.h"
#include "coap_log.h"
```

## Macros

- `#define COAP_CLIENT_ACK_TIMEOUT_SEC 2`
- `#define COAP_CLIENT_MAX_RETRANSMIT 4`
- `#define COAP_CLIENT_RESP_TIMEOUT_SEC 30`

## Functions

- `int coap_client_create (coap_client_t *client, const char *host, unsigned port)`  
*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.*

### 5.5.1 Detailed Description

Source file for the FreeCoAP client library.

### 5.5.2 Macro Definition Documentation

#### 5.5.2.1 `#define COAP_CLIENT_ACK_TIMEOUT_SEC 2`

Minimum delay to wait before retransmitting a confirmable message

#### 5.5.2.2 `#define COAP_CLIENT_MAX_RETRANSMIT 4`

Maximum number of times a confirmable message can be retransmitted

#### 5.5.2.3 `#define COAP_CLIENT_RESP_TIMEOUT_SEC 30`

Maximum amount of time to wait for a response

### 5.5.3 Function Documentation

#### 5.5.3.1 `int coap_client_create ( coap_client_t * client, const char * host, unsigned port )`

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

**Returns**

Operation status

**Return values**

0	Success
-errno	On error

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

Deinitialise a client structure.

**Parameters**

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

**5.5.3.3 int coap\_client\_exchange ( coap\_client\_t \* client, coap\_msg\_t \* req, coap\_msg\_t \* resp )**

Send a request to the server and receive the response.

**Parameters**

in, out	<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

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

**Returns**

Operation status

**Return values**

0	Success
-errno	Error

**5.6 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"
```

## 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, unsigned 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.*
- int `coap_msg_parse_type_msg_id` (char \*buf, unsigned len, unsigned \*type, unsigned \*msg\_id)  
*Extract the type and message ID values from a message.*
- int `coap_msg_parse` (coap\_msg\_t \*msg, char \*buf, unsigned 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, unsigned len)  
*Set the token in a message.*
- int `coap_msg_add_op` (coap\_msg\_t \*msg, unsigned num, unsigned len, char \*val)  
*Add a token to a message structure.*
- int `coap_msg_set_payload` (coap\_msg\_t \*msg, char \*buf, unsigned len)  
*Set the payload in a message.*
- int `coap_msg_format` (coap\_msg\_t \*msg, char \*buf, unsigned len)  
*Format a message.*
- int `coap_msg_copy` (coap\_msg\_t \*dst, coap\_msg\_t \*src)  
*Copy a message.*

### 5.6.1 Detailed Description

Source file for the FreeCoAP message parser/formatter library.

### 5.6.2 Macro Definition Documentation

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

Get the first option from an option linked-list

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

Get the last option in an option linked-list



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

Indicate if an option linked-list is empty

### 5.6.3 Function Documentation

5.6.3.1 `int coap_msg_add_op ( coap_msg_t * msg, unsigned num, unsigned len, char * val )`

Add a token to a message structure.

#### Parameters

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
-ENOMEM	Out-of-memory

5.6.3.2 `int coap_msg_copy ( coap_msg_t * dst, coap_msg_t * src )`

Copy a message.

#### Parameters

out	<i>dst</i>	Pointer to the destination message structure
in	<i>src</i>	Pointer to the source message structure

#### Returns

Operation status

#### Return values

0	Success
-EINVAL	Invalid argument
-ENOMEM	Out-of-memory

5.6.3.3 `void coap_msg_create ( coap_msg_t * msg )`

Initialise a message structure.

#### Parameters

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

5.6.3.4 `void coap_msg_destroy ( coap_msg_t * msg )`

Deinitialise a message structure.

## Parameters

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

5.6.3.5 int coap\_msg\_format ( coap\_msg\_t \* *msg*, char \* *buf*, unsigned *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

Operation status

## Return values

> 0	Length of the formatted message
-ENOSPC	Insufficient buffer length
-EBADMSG	Message is corrupt

5.6.3.6 void coap\_msg\_gen\_rand\_str ( char \* *buf*, unsigned *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

5.6.3.7 int coap\_msg\_parse ( coap\_msg\_t \* *msg*, char \* *buf*, unsigned *len* )

Parse a message.

## Parameters

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
-EINVAL	Invalid argument
-ENOMEM	Out-of-memory

-EBADMSG	The message is corrupt
----------	------------------------

**5.6.3.8** `int coap_msg_parse_type_msg_id ( char * buf, unsigned 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
-EBADMSG	The message is corrupt

**5.6.3.9** `void coap_msg_reset ( coap_msg_t * msg )`

Deinitialise and initialise a message structure.

#### Parameters

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

**5.6.3.10** `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
-EINVAL	Invalid argument

**5.6.3.11** `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_id</i>	Message ID

**Returns**

Operation status

**Return values**

0	Success
-EINVAL	Invalid argument

**5.6.3.12 int coap\_msg\_set\_payload ( coap\_msg\_t \* msg, char \* buf, unsigned 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**

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
-ENOMEM	Out-of-memory

**5.6.3.13 int coap\_msg\_set\_token ( coap\_msg\_t \* msg, char \* buf, unsigned 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
-EINVAL	Invalid argument

**5.6.3.14 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
-EINVAL	Invalid argument

## 5.7 src/coap\_server.c File Reference

Source file for the FreeCoAP server library.

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

### Macros

- `#define COAP_SERVER_ACK_TIMEOUT_SEC 2`
- `#define COAP_SERVER_MAX_RETRANSMIT 4`

### Functions

- `int coap_server_create (coap_server_t *server, int(*handle)(coap_server_t *, coap_msg_t *, coap_msg_t *), const char *host, unsigned port)`  
*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.*

### 5.7.1 Detailed Description

Source file for the FreeCoAP server library.

### 5.7.2 Macro Definition Documentation

#### 5.7.2.1 `#define COAP_SERVER_ACK_TIMEOUT_SEC 2`

Minimum delay to wait before retransmitting a confirmable message

#### 5.7.2.2 `#define COAP_SERVER_MAX_RETRANSMIT 4`

Maximum number of times a confirmable message can be retransmitted

### 5.7.3 Function Documentation

#### 5.7.3.1 `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	<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
-ENOMEM	Out-of-memory

#### 5.7.3.2 `int coap_server_create ( coap_server_t * server, int (*)(coap_server_t *, coap_msg_t *, coap_msg_t *) handle, const char * host, unsigned port )`

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

Returns

Operation status

Return values

<i>0</i>	Success
<i>-errno</i>	Error

#### 5.7.3.3 void coap\_server\_destroy ( coap\_server\_t \* server )

Deinitialise a server structure.

##### Parameters

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

#### 5.7.3.4 unsigned coap\_server\_get\_next\_msg\_id ( coap\_server\_t \* server )

Get a new message ID value.

##### Parameters

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

##### Returns

message ID value

#### 5.7.3.5 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.

##### Returns

Operation status

##### Return values

<i>0</i>	Success
<i>-errno</i>	Error





# Index

COAP\_CLIENT\_ACK\_TIMEOUT\_SEC  
    coap\_client.c, 30

COAP\_CLIENT\_ADDR\_BUF\_LEN  
    coap\_client.h, 13

COAP\_CLIENT\_MAX\_RETRANSMIT  
    coap\_client.c, 30

COAP\_CLIENT\_RESP\_TIMEOUT\_SEC  
    coap\_client.c, 30

COAP\_LOG\_DEBUG  
    coap\_log.h, 16

COAP\_LOG\_DEF\_LEVEL  
    coap\_log.h, 15

COAP\_LOG\_ERROR  
    coap\_log.h, 16

COAP\_MSG\_MAX\_BUF\_LEN  
    coap\_msg.h, 20

COAP\_MSG\_MAX\_CODE\_CLASS  
    coap\_msg.h, 20

COAP\_MSG\_MAX\_CODE\_DETAIL  
    coap\_msg.h, 20

COAP\_MSG\_MAX\_MSG\_ID  
    coap\_msg.h, 20

COAP\_MSG\_MAX\_TOKEN\_LEN  
    coap\_msg.h, 20

COAP\_MSG\_OP\_URI\_PATH\_MAX\_LEN  
    coap\_msg.h, 21

COAP\_MSG\_OP\_URI\_PATH\_NUM  
    coap\_msg.h, 21

COAP\_MSG\_VER  
    coap\_msg.h, 21

COAP\_SERVER\_ACK\_TIMEOUT\_SEC  
    coap\_server.c, 38

COAP\_SERVER\_ADDR\_BUF\_LEN  
    coap\_server.h, 28

COAP\_SERVER\_MAX\_RETRANSMIT  
    coap\_server.c, 38

COAP\_SERVER\_NUM\_TRANS  
    coap\_server.h, 28

coap\_client.c  
    COAP\_CLIENT\_ACK\_TIMEOUT\_SEC, 30  
    COAP\_CLIENT\_MAX\_RETRANSMIT, 30  
    COAP\_CLIENT\_RESP\_TIMEOUT\_SEC, 30  
    coap\_client\_create, 30  
    coap\_client\_destroy, 31  
    coap\_client\_exchange, 31

coap\_client.h  
    COAP\_CLIENT\_ADDR\_BUF\_LEN, 13  
    coap\_client\_create, 14  
    coap\_client\_destroy, 14  
    coap\_client\_exchange, 14  
    coap\_client\_create  
        coap\_client.c, 30  
        coap\_client.h, 14  
    coap\_client\_destroy  
        coap\_client.c, 31  
        coap\_client.h, 14  
    coap\_client\_exchange  
        coap\_client.c, 31  
        coap\_client.h, 14  
    coap\_client\_t, 7  
    coap\_log.h  
        COAP\_LOG\_DEBUG, 16  
        COAP\_LOG\_DEF\_LEVEL, 15  
        COAP\_LOG\_ERROR, 16  
        coap\_log\_debug, 16  
        coap\_log\_error, 16  
        coap\_log\_info, 16  
        coap\_log\_level\_t, 16  
        coap\_log\_notice, 16  
        coap\_log\_set\_level, 16  
        coap\_log\_warn, 17  
    coap\_log\_debug  
        coap\_log.h, 16  
    coap\_log\_error  
        coap\_log.h, 16  
    coap\_log\_info  
        coap\_log.h, 16  
    coap\_log\_level\_t  
        coap\_log.h, 16  
    coap\_log\_notice  
        coap\_log.h, 16  
    coap\_log\_set\_level  
        coap\_log.h, 16  
    coap\_log\_warn  
        coap\_log.h, 17  
    coap\_msg.c  
        coap\_msg\_add\_op, 33  
        coap\_msg\_copy, 33  
        coap\_msg\_create, 33  
        coap\_msg\_destroy, 33  
        coap\_msg\_format, 34  
        coap\_msg\_gen\_rand\_str, 34  
        coap\_msg\_op\_list\_get\_first, 32  
        coap\_msg\_op\_list\_get\_last, 32  
        coap\_msg\_op\_list\_is\_empty, 32  
        coap\_msg\_parse, 34  
        coap\_msg\_parse\_type\_msg\_id, 35  
        coap\_msg\_reset, 35

- coap\_msg\_set\_code, 35
- coap\_msg\_set\_msg\_id, 35
- coap\_msg\_set\_payload, 36
- coap\_msg\_set\_token, 36
- coap\_msg\_set\_type, 36
- coap\_msg.h
  - COAP\_MSG\_MAX\_BUF\_LEN, 20
  - COAP\_MSG\_MAX\_CODE\_CLASS, 20
  - COAP\_MSG\_MAX\_CODE\_DETAIL, 20
  - COAP\_MSG\_MAX\_MSG\_ID, 20
  - COAP\_MSG\_MAX\_TOKEN\_LEN, 20
  - COAP\_MSG\_OP\_URI\_PATH\_MAX\_LEN, 21
  - COAP\_MSG\_OP\_URI\_PATH\_NUM, 21
  - COAP\_MSG\_VER, 21
  - coap\_msg\_add\_op, 22
  - coap\_msg\_copy, 23
  - coap\_msg\_create, 23
  - coap\_msg\_destroy, 23
  - coap\_msg\_format, 23
  - coap\_msg\_gen\_rand\_str, 24
  - coap\_msg\_get\_code\_class, 19
  - coap\_msg\_get\_code\_detail, 19
  - coap\_msg\_get\_first\_op, 19
  - coap\_msg\_get\_msg\_id, 19
  - coap\_msg\_get\_payload, 19
  - coap\_msg\_get\_payload\_len, 19
  - coap\_msg\_get\_token, 19
  - coap\_msg\_get\_token\_len, 20
  - coap\_msg\_get\_type, 20
  - coap\_msg\_get\_ver, 20
  - coap\_msg\_is\_empty, 20
  - coap\_msg\_op\_get\_len, 20
  - coap\_msg\_op\_get\_next, 20
  - coap\_msg\_op\_get\_num, 20
  - coap\_msg\_op\_get\_val, 21
  - coap\_msg\_op\_num\_is\_critical, 21
  - coap\_msg\_op\_num\_is\_unsafe, 21
  - coap\_msg\_op\_num\_no\_cache\_key, 21
  - coap\_msg\_op\_set\_len, 21
  - coap\_msg\_op\_set\_next, 21
  - coap\_msg\_op\_set\_num, 21
  - coap\_msg\_op\_set\_val, 21
  - coap\_msg\_parse, 24
  - coap\_msg\_parse\_type\_msg\_id, 24
  - coap\_msg\_reset, 25
  - coap\_msg\_set\_code, 25
  - coap\_msg\_set\_msg\_id, 25
  - coap\_msg\_set\_payload, 26
  - coap\_msg\_set\_token, 26
  - coap\_msg\_set\_type, 26
- coap\_msg\_add\_op
  - coap\_msg.c, 33
  - coap\_msg.h, 22
- coap\_msg\_copy
  - coap\_msg.c, 33
  - coap\_msg.h, 23
- coap\_msg\_create
  - coap\_msg.c, 33
- coap\_msg.h, 23
- coap\_msg\_destroy
  - coap\_msg.c, 33
  - coap\_msg.h, 23
- coap\_msg\_format
  - coap\_msg.c, 34
  - coap\_msg.h, 23
- coap\_msg\_gen\_rand\_str
  - coap\_msg.c, 34
  - coap\_msg.h, 24
- coap\_msg\_get\_code\_class
  - coap\_msg.h, 19
- coap\_msg\_get\_code\_detail
  - coap\_msg.h, 19
- coap\_msg\_get\_first\_op
  - coap\_msg.h, 19
- coap\_msg\_get\_msg\_id
  - coap\_msg.h, 19
- coap\_msg\_get\_payload
  - coap\_msg.h, 19
- coap\_msg\_get\_payload\_len
  - coap\_msg.h, 19
- coap\_msg\_get\_token
  - coap\_msg.h, 19
- coap\_msg\_get\_token\_len
  - coap\_msg.h, 20
- coap\_msg\_get\_type
  - coap\_msg.h, 20
- coap\_msg\_get\_ver
  - coap\_msg.h, 20
- coap\_msg\_is\_empty
  - coap\_msg.h, 20
- coap\_msg\_op, 7
- coap\_msg\_op\_get\_len
  - coap\_msg.h, 20
- coap\_msg\_op\_get\_next
  - coap\_msg.h, 20
- coap\_msg\_op\_get\_num
  - coap\_msg.h, 20
- coap\_msg\_op\_get\_val
  - coap\_msg.h, 21
- coap\_msg\_op\_list\_get\_first
  - coap\_msg.c, 32
- coap\_msg\_op\_list\_get\_last
  - coap\_msg.c, 32
- coap\_msg\_op\_list\_is\_empty
  - coap\_msg.c, 32
- coap\_msg\_op\_list\_t, 8
- coap\_msg\_op\_num\_is\_critical
  - coap\_msg.h, 21
- coap\_msg\_op\_num\_is\_unsafe
  - coap\_msg.h, 21
- coap\_msg\_op\_num\_no\_cache\_key
  - coap\_msg.h, 21
- coap\_msg\_op\_set\_len
  - coap\_msg.h, 21
- coap\_msg\_op\_set\_next
  - coap\_msg.h, 21

- coap\_msg\_op\_set\_num
  - coap\_msg.h, [21](#)
- coap\_msg\_op\_set\_val
  - coap\_msg.h, [21](#)
- coap\_msg\_parse
  - coap\_msg.c, [34](#)
  - coap\_msg.h, [24](#)
- coap\_msg\_parse\_type\_msg\_id
  - coap\_msg.c, [35](#)
  - coap\_msg.h, [24](#)
- coap\_msg\_reset
  - coap\_msg.c, [35](#)
  - coap\_msg.h, [25](#)
- coap\_msg\_set\_code
  - coap\_msg.c, [35](#)
  - coap\_msg.h, [25](#)
- coap\_msg\_set\_msg\_id
  - coap\_msg.c, [35](#)
  - coap\_msg.h, [25](#)
- coap\_msg\_set\_payload
  - coap\_msg.c, [36](#)
  - coap\_msg.h, [26](#)
- coap\_msg\_set\_token
  - coap\_msg.c, [36](#)
  - coap\_msg.h, [26](#)
- coap\_msg\_set\_type
  - coap\_msg.c, [36](#)
  - coap\_msg.h, [26](#)
- coap\_msg\_t, [8](#)
- coap\_server, [9](#)
- coap\_server.c
  - COAP\_SERVER\_ACK\_TIMEOUT\_SEC, [38](#)
  - COAP\_SERVER\_MAX\_RETRANSMIT, [38](#)
  - coap\_server\_add\_sep\_resp\_uri\_path, [38](#)
  - coap\_server\_create, [38](#)
  - coap\_server\_destroy, [39](#)
  - coap\_server\_get\_next\_msg\_id, [39](#)
  - coap\_server\_run, [39](#)
- coap\_server.h
  - COAP\_SERVER\_ADDR\_BUF\_LEN, [28](#)
  - COAP\_SERVER\_NUM\_TRANS, [28](#)
  - coap\_server\_add\_sep\_resp\_uri\_path, [28](#)
  - coap\_server\_create, [28](#)
  - coap\_server\_destroy, [29](#)
  - coap\_server\_get\_next\_msg\_id, [29](#)
  - coap\_server\_run, [29](#)
- coap\_server\_add\_sep\_resp\_uri\_path
  - coap\_server.c, [38](#)
  - coap\_server.h, [28](#)
- coap\_server\_create
  - coap\_server.c, [38](#)
  - coap\_server.h, [28](#)
- coap\_server\_destroy
  - coap\_server.c, [39](#)
  - coap\_server.h, [29](#)
- coap\_server\_get\_next\_msg\_id
  - coap\_server.c, [39](#)
  - coap\_server.h, [29](#)
- coap\_server\_path, [9](#)
- coap\_server\_path\_list\_t, [9](#)
- coap\_server\_run
  - coap\_server.c, [39](#)
  - coap\_server.h, [29](#)
- coap\_server\_trans, [10](#)
- harness\_test, [10](#)
- include/coap\_client.h, [13](#)
- include/coap\_log.h, [15](#)
- include/coap\_msg.h, [17](#)
- include/coap\_server.h, [27](#)
- src/coap\_client.c, [29](#)
- src/coap\_msg.c, [31](#)
- src/coap\_server.c, [37](#)
- test\_data, [10](#)
- test\_op, [11](#)