

FreeCoAP

Generated by Doxygen 1.8.11

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
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_mem_t Struct Reference	7
3.2.1	Detailed Description	7
3.2.2	Member Data Documentation	7

3.2.2.1	active	7
3.2.2.2	buf	7
3.2.2.3	len	7
3.2.2.4	num	7
3.3	coap_msg_op Struct Reference	8
3.3.1	Detailed Description	8
3.3.2	Member Data Documentation	8
3.3.2.1	len	8
3.3.2.2	next	8
3.3.2.3	num	8
3.3.2.4	val	9
3.4	coap_msg_op_list_t Struct Reference	9
3.4.1	Detailed Description	9
3.4.2	Member Data Documentation	9
3.4.2.1	first	9
3.4.2.2	last	10
3.5	coap_msg_t Struct Reference	10
3.5.1	Detailed Description	11
3.5.2	Member Data Documentation	11
3.5.2.1	code_class	11
3.5.2.2	code_detail	11
3.5.2.3	msg_id	11
3.5.2.4	op_list	11
3.5.2.5	payload	11
3.5.2.6	payload_len	11
3.5.2.7	token	11
3.5.2.8	token_len	11
3.5.2.9	type	11
3.5.2.10	ver	12
3.6	coap_server Struct Reference	12

3.6.1	Detailed Description	13
3.6.2	Member Data Documentation	13
3.6.2.1	cred	13
3.6.2.2	dh_params	13
3.6.2.3	handle	13
3.6.2.4	msg_id	13
3.6.2.5	priority	13
3.6.2.6	sd	13
3.6.2.7	sep_list	13
3.6.2.8	trans	13
3.7	coap_server_path Struct Reference	14
3.7.1	Detailed Description	14
3.7.2	Member Data Documentation	14
3.7.2.1	next	14
3.7.2.2	str	14
3.8	coap_server_path_list_t Struct Reference	15
3.8.1	Detailed Description	15
3.8.2	Member Data Documentation	15
3.8.2.1	first	15
3.8.2.2	last	15
3.9	coap_server_trans Struct Reference	16
3.9.1	Detailed Description	17
3.9.2	Member Data Documentation	17
3.9.2.1	active	17
3.9.2.2	block1_next	17
3.9.2.3	block1_size	17
3.9.2.4	block2_next	17
3.9.2.5	block2_size	17
3.9.2.6	block_detail	17
3.9.2.7	block_rx	18

3.9.2.8	block_uri	18
3.9.2.9	body	18
3.9.2.10	body_end	18
3.9.2.11	body_len	18
3.9.2.12	client_addr	18
3.9.2.13	client_sin	18
3.9.2.14	client_sin_len	18
3.9.2.15	last_use	18
3.9.2.16	num_retrans	18
3.9.2.17	req	19
3.9.2.18	resp	19
3.9.2.19	server	19
3.9.2.20	session	19
3.9.2.21	timeout	19
3.9.2.22	timer_fd	19
3.9.2.23	type	19
4	File Documentation	21
4.1	lib/include/coap_client.h File Reference	21
4.1.1	Detailed Description	22
4.1.2	Macro Definition Documentation	22
4.1.2.1	COAP_CLIENT_HOST_BUF_LEN	22
4.1.2.2	COAP_CLIENT_PORT_BUF_LEN	22
4.1.3	Function Documentation	22
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)	22
4.1.3.2	coap_client_destroy(coap_client_t *client)	23
4.1.3.3	coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)	23
4.1.3.4	coap_client_exchange_blockwise(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, int have_resp)	24
4.2	lib/include/coap_ipv.h File Reference	24

4.2.1	Detailed Description	25
4.3	lib/include/coap_log.h File Reference	26
4.3.1	Detailed Description	26
4.3.2	Macro Definition Documentation	27
4.3.2.1	COAP_LOG_DEF_LEVEL	27
4.3.3	Enumeration Type Documentation	27
4.3.3.1	coap_log_level_t	27
4.3.4	Function Documentation	27
4.3.4.1	coap_log_debug(const char *msg,...)	27
4.3.4.2	coap_log_error(const char *msg,...)	27
4.3.4.3	coap_log_get_level(void)	27
4.3.4.4	coap_log_info(const char *msg,...)	28
4.3.4.5	coap_log_notice(const char *msg,...)	28
4.3.4.6	coap_log_set_level(coap_log_level_t level)	28
4.3.4.7	coap_log_warn(const char *msg,...)	28
4.4	lib/include/coap_mem.h File Reference	29
4.4.1	Detailed Description	31
4.4.2	Macro Definition Documentation	31
4.4.2.1	coap_mem_get_active	31
4.4.2.2	coap_mem_get_active_len	31
4.4.2.3	coap_mem_get_buf	31
4.4.2.4	coap_mem_get_len	31
4.4.2.5	coap_mem_get_num	31
4.4.3	Function Documentation	31
4.4.3.1	coap_mem_all_create(size_t small_num, size_t small_len, size_t medium_num, size_t medium_len, size_t large_num, size_t large_len)	31
4.4.3.2	coap_mem_alloc(coap_mem_t *mem, size_t len)	32
4.4.3.3	coap_mem_create(coap_mem_t *mem, size_t num, size_t len)	32
4.4.3.4	coap_mem_destroy(coap_mem_t *mem)	33
4.4.3.5	coap_mem_free(coap_mem_t *mem, void *buf)	33
4.4.3.6	coap_mem_large_alloc(size_t len)	33

4.4.3.7	<code>coap_mem_large_create(size_t num, size_t len)</code>	33
4.4.3.8	<code>coap_mem_large_free(void *buf)</code>	34
4.4.3.9	<code>coap_mem_large_get_active(void)</code>	34
4.4.3.10	<code>coap_mem_large_get_active_len(void)</code>	34
4.4.3.11	<code>coap_mem_large_get_buf(void)</code>	34
4.4.3.12	<code>coap_mem_large_get_len(void)</code>	35
4.4.3.13	<code>coap_mem_large_get_num(void)</code>	35
4.4.3.14	<code>coap_mem_medium_alloc(size_t len)</code>	35
4.4.3.15	<code>coap_mem_medium_create(size_t num, size_t len)</code>	35
4.4.3.16	<code>coap_mem_medium_free(void *buf)</code>	36
4.4.3.17	<code>coap_mem_medium_get_active(void)</code>	36
4.4.3.18	<code>coap_mem_medium_get_active_len(void)</code>	36
4.4.3.19	<code>coap_mem_medium_get_buf(void)</code>	36
4.4.3.20	<code>coap_mem_medium_get_len(void)</code>	36
4.4.3.21	<code>coap_mem_medium_get_num(void)</code>	36
4.4.3.22	<code>coap_mem_small_alloc(size_t len)</code>	36
4.4.3.23	<code>coap_mem_small_create(size_t num, size_t len)</code>	37
4.4.3.24	<code>coap_mem_small_free(void *buf)</code>	37
4.4.3.25	<code>coap_mem_small_get_active(void)</code>	37
4.4.3.26	<code>coap_mem_small_get_active_len(void)</code>	38
4.4.3.27	<code>coap_mem_small_get_buf(void)</code>	38
4.4.3.28	<code>coap_mem_small_get_len(void)</code>	38
4.4.3.29	<code>coap_mem_small_get_num(void)</code>	38
4.5	<code>lib/include/coap_msg.h</code> File Reference	38
4.5.1	Detailed Description	42
4.5.2	Macro Definition Documentation	42
4.5.2.1	<code>coap_msg_block_start_to_num</code>	42
4.5.2.2	<code>coap_msg_block_szx_to_size</code>	42
4.5.2.3	<code>coap_msg_get_code_class</code>	42
4.5.2.4	<code>coap_msg_get_code_detail</code>	42

4.5.2.5	coap_msg_get_first_op	42
4.5.2.6	coap_msg_get_msg_id	42
4.5.2.7	coap_msg_get_payload	42
4.5.2.8	coap_msg_get_payload_len	42
4.5.2.9	coap_msg_get_token	42
4.5.2.10	coap_msg_get_token_len	43
4.5.2.11	coap_msg_get_type	43
4.5.2.12	coap_msg_get_ver	43
4.5.2.13	coap_msg_is_empty	43
4.5.2.14	COAP_MSG_MAX_BUF_LEN	43
4.5.2.15	COAP_MSG_MAX_CODE_CLASS	43
4.5.2.16	COAP_MSG_MAX_CODE_DETAIL	43
4.5.2.17	COAP_MSG_MAX_MSG_ID	43
4.5.2.18	COAP_MSG_MAX_PAYLOAD_LEN	43
4.5.2.19	COAP_MSG_MAX_TOKEN_LEN	43
4.5.2.20	coap_msg_op_get_len	44
4.5.2.21	coap_msg_op_get_next	44
4.5.2.22	coap_msg_op_get_num	44
4.5.2.23	coap_msg_op_get_val	44
4.5.2.24	COAP_MSG_OP_MAX_BLOCK_SIZE	44
4.5.2.25	COAP_MSG_OP_MAX_BLOCK_VAL_LEN	44
4.5.2.26	coap_msg_op_num_is_critical	44
4.5.2.27	coap_msg_op_num_is_unsafe	44
4.5.2.28	coap_msg_op_num_no_cache_key	44
4.5.2.29	coap_msg_op_set_len	44
4.5.2.30	coap_msg_op_set_next	45
4.5.2.31	coap_msg_op_set_num	45
4.5.2.32	coap_msg_op_set_val	45
4.5.2.33	COAP_MSG_OP_URI_PATH_MAX_LEN	45
4.5.2.34	COAP_MSG_VER	45

4.5.3	Enumeration Type Documentation	45
4.5.3.1	coap_msg_class_t	45
4.5.3.2	coap_msg_client_err_t	45
4.5.3.3	coap_msg_method_t	46
4.5.3.4	coap_msg_op_num_t	46
4.5.3.5	coap_msg_server_err_t	46
4.5.3.6	coap_msg_success_t	47
4.5.3.7	coap_msg_type_t	47
4.5.4	Function Documentation	47
4.5.4.1	coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, const char *val)	47
4.5.4.2	coap_msg_check_critical_ops(coap_msg_t *msg)	48
4.5.4.3	coap_msg_check_unsafe_ops(coap_msg_t *msg)	48
4.5.4.4	coap_msg_clear_payload(coap_msg_t *msg)	48
4.5.4.5	coap_msg_copy(coap_msg_t *dst, coap_msg_t *src)	49
4.5.4.6	coap_msg_create(coap_msg_t *msg)	49
4.5.4.7	coap_msg_destroy(coap_msg_t *msg)	49
4.5.4.8	coap_msg_format(coap_msg_t *msg, char *buf, size_t len)	49
4.5.4.9	coap_msg_gen_rand_str(char *buf, size_t len)	50
4.5.4.10	coap_msg_op_calc_block_szx(unsigned size)	50
4.5.4.11	coap_msg_op_format_block_val(char *val, unsigned len, unsigned num, unsigned more, unsigned size)	50
4.5.4.12	coap_msg_op_num_is_recognized(unsigned num)	51
4.5.4.13	coap_msg_op_parse_block_val(unsigned *num, unsigned *more, unsigned *size, const char *val, unsigned len)	51
4.5.4.14	coap_msg_parse(coap_msg_t *msg, char *buf, size_t len)	52
4.5.4.15	coap_msg_parse_block_op(unsigned *num, unsigned *more, unsigned *size, coap_msg_t *msg, int type)	52
4.5.4.16	coap_msg_parse_type_msg_id(char *buf, size_t len, unsigned *type, unsigned *msg_id)	52
4.5.4.17	coap_msg_reset(coap_msg_t *msg)	53
4.5.4.18	coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code↔_detail)	53

4.5.4.19	<code>coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id)</code>	54
4.5.4.20	<code>coap_msg_set_payload(coap_msg_t *msg, char *buf, size_t len)</code>	54
4.5.4.21	<code>coap_msg_set_token(coap_msg_t *msg, char *buf, size_t len)</code>	54
4.5.4.22	<code>coap_msg_set_type(coap_msg_t *msg, unsigned type)</code>	55
4.5.4.23	<code>coap_msg_uri_path_to_str(coap_msg_t *msg, char *buf, size_t len)</code>	55
4.6	<code>lib/include/coap_server.h</code> File Reference	56
4.6.1	Detailed Description	58
4.6.2	Macro Definition Documentation	58
4.6.2.1	<code>COAP_SERVER_ADDR_BUF_LEN</code>	58
4.6.2.2	<code>COAP_SERVER_DIAG_PAYLOAD_LEN</code>	58
4.6.2.3	<code>COAP_SERVER_NUM_TRANS</code>	58
4.6.2.4	<code>coap_server_trans_get_body</code>	58
4.6.2.5	<code>coap_server_trans_get_body_end</code>	58
4.6.2.6	<code>coap_server_trans_get_body_len</code>	58
4.6.2.7	<code>coap_server_trans_get_req</code>	58
4.6.2.8	<code>coap_server_trans_get_resp</code>	58
4.6.2.9	<code>coap_server_trans_get_type</code>	58
4.6.2.10	<code>coap_server_trans_set_body_end</code>	58
4.6.3	Typedef Documentation	58
4.6.3.1	<code>coap_server_trans_handler_t</code>	58
4.6.4	Enumeration Type Documentation	59
4.6.4.1	<code>coap_server_resp_t</code>	59
4.6.4.2	<code>coap_server_trans_type_t</code>	59
4.6.5	Function Documentation	59
4.6.5.1	<code>coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str)</code> . .	59
4.6.5.2	<code>coap_server_create(coap_server_t *server, coap_server_trans_handler_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)</code>	60
4.6.5.3	<code>coap_server_destroy(coap_server_t *server)</code>	60
4.6.5.4	<code>coap_server_get_next_msg_id(coap_server_t *server)</code>	61
4.6.5.5	<code>coap_server_run(coap_server_t *server)</code>	61

4.6.5.6	coap_server_trans_handle_blockwise(coap_server_trans_t *trans, coap_msg↵ _t *req, coap_msg_t *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, coap_server_trans_handler_t block_rx)	61
4.7	lib/src/coap_client.c File Reference	62
4.7.1	Detailed Description	63
4.7.2	Macro Definition Documentation	63
4.7.2.1	COAP_CLIENT_ACK_TIMEOUT_SEC	63
4.7.2.2	COAP_CLIENT_DTLS_HANDSHAKE_ATTEMPTS	63
4.7.2.3	COAP_CLIENT_DTLS_MTU	63
4.7.2.4	COAP_CLIENT_DTLS_PRIORITIES	63
4.7.2.5	COAP_CLIENT_DTLS_RETRANS_TIMEOUT	64
4.7.2.6	COAP_CLIENT_DTLS_TOTAL_TIMEOUT	64
4.7.2.7	COAP_CLIENT_MAX_RETRANSMIT	64
4.7.2.8	COAP_CLIENT_RESP_TIMEOUT_SEC	64
4.7.3	Function Documentation	64
4.7.3.1	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)	64
4.7.3.2	coap_client_destroy(coap_client_t *client)	65
4.7.3.3	coap_client_exchange(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp)	65
4.7.3.4	coap_client_exchange_blockwise(coap_client_t *client, coap_msg_t *req, coap_msg_t *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, int have_resp)	65
4.8	lib/src/coap_log.c File Reference	66
4.8.1	Detailed Description	67
4.8.2	Function Documentation	67
4.8.2.1	coap_log_debug(const char *msg,...)	67
4.8.2.2	coap_log_error(const char *msg,...)	67
4.8.2.3	coap_log_get_level(void)	67
4.8.2.4	coap_log_info(const char *msg,...)	67
4.8.2.5	coap_log_notice(const char *msg,...)	68
4.8.2.6	coap_log_set_level(coap_log_level_t level)	68
4.8.2.7	coap_log_warn(const char *msg,...)	68

4.9	lib/src/coap_mem.c File Reference	68
4.9.1	Detailed Description	70
4.9.2	Function Documentation	70
4.9.2.1	coap_mem_all_create(size_t small_num, size_t small_len, size_t medium_num, size_t medium_len, size_t large_num, size_t large_len)	70
4.9.2.2	coap_mem_alloc(coap_mem_t *mem, size_t len)	71
4.9.2.3	coap_mem_create(coap_mem_t *mem, size_t num, size_t len)	71
4.9.2.4	coap_mem_destroy(coap_mem_t *mem)	72
4.9.2.5	coap_mem_free(coap_mem_t *mem, void *buf)	72
4.9.2.6	coap_mem_large_alloc(size_t len)	72
4.9.2.7	coap_mem_large_create(size_t num, size_t len)	72
4.9.2.8	coap_mem_large_free(void *buf)	73
4.9.2.9	coap_mem_large_get_active(void)	73
4.9.2.10	coap_mem_large_get_active_len(void)	73
4.9.2.11	coap_mem_large_get_buf(void)	73
4.9.2.12	coap_mem_large_get_len(void)	74
4.9.2.13	coap_mem_large_get_num(void)	74
4.9.2.14	coap_mem_medium_alloc(size_t len)	74
4.9.2.15	coap_mem_medium_create(size_t num, size_t len)	74
4.9.2.16	coap_mem_medium_free(void *buf)	75
4.9.2.17	coap_mem_medium_get_active(void)	75
4.9.2.18	coap_mem_medium_get_active_len(void)	75
4.9.2.19	coap_mem_medium_get_buf(void)	75
4.9.2.20	coap_mem_medium_get_len(void)	75
4.9.2.21	coap_mem_medium_get_num(void)	75
4.9.2.22	coap_mem_small_alloc(size_t len)	75
4.9.2.23	coap_mem_small_create(size_t num, size_t len)	76
4.9.2.24	coap_mem_small_free(void *buf)	76
4.9.2.25	coap_mem_small_get_active(void)	76
4.9.2.26	coap_mem_small_get_active_len(void)	77
4.9.2.27	coap_mem_small_get_buf(void)	77

4.9.2.28	<code>coap_mem_small_get_len(void)</code>	77
4.9.2.29	<code>coap_mem_small_get_num(void)</code>	77
4.10	<code>lib/src/coap_msg.c</code> File Reference	77
4.10.1	Detailed Description	79
4.10.2	Macro Definition Documentation	79
4.10.2.1	<code>coap_msg_op_list_get_first</code>	79
4.10.2.2	<code>coap_msg_op_list_get_last</code>	79
4.10.2.3	<code>coap_msg_op_list_is_empty</code>	79
4.10.3	Function Documentation	79
4.10.3.1	<code>coap_msg_add_op(coap_msg_t *msg, unsigned num, unsigned len, const char *val)</code>	79
4.10.3.2	<code>coap_msg_check_critical_ops(coap_msg_t *msg)</code>	80
4.10.3.3	<code>coap_msg_check_unsafe_ops(coap_msg_t *msg)</code>	80
4.10.3.4	<code>coap_msg_clear_payload(coap_msg_t *msg)</code>	80
4.10.3.5	<code>coap_msg_copy(coap_msg_t *dst, coap_msg_t *src)</code>	81
4.10.3.6	<code>coap_msg_create(coap_msg_t *msg)</code>	81
4.10.3.7	<code>coap_msg_destroy(coap_msg_t *msg)</code>	81
4.10.3.8	<code>coap_msg_format(coap_msg_t *msg, char *buf, size_t len)</code>	81
4.10.3.9	<code>coap_msg_gen_rand_str(char *buf, size_t len)</code>	82
4.10.3.10	<code>coap_msg_op_calc_block_szx(unsigned size)</code>	82
4.10.3.11	<code>coap_msg_op_format_block_val(char *val, unsigned len, unsigned num, unsigned more, unsigned size)</code>	82
4.10.3.12	<code>coap_msg_op_num_is_recognized(unsigned num)</code>	83
4.10.3.13	<code>coap_msg_op_parse_block_val(unsigned *num, unsigned *more, unsigned *size, const char *val, unsigned len)</code>	83
4.10.3.14	<code>coap_msg_parse(coap_msg_t *msg, char *buf, size_t len)</code>	84
4.10.3.15	<code>coap_msg_parse_block_op(unsigned *num, unsigned *more, unsigned *size, coap_msg_t *msg, int type)</code>	84
4.10.3.16	<code>coap_msg_parse_type_msg_id(char *buf, size_t len, unsigned *type, unsigned *msg_id)</code>	84
4.10.3.17	<code>coap_msg_reset(coap_msg_t *msg)</code>	85
4.10.3.18	<code>coap_msg_set_code(coap_msg_t *msg, unsigned code_class, unsigned code↔_detail)</code>	85

4.10.3.19	<code>coap_msg_set_msg_id(coap_msg_t *msg, unsigned msg_id)</code>	86
4.10.3.20	<code>coap_msg_set_payload(coap_msg_t *msg, char *buf, size_t len)</code>	86
4.10.3.21	<code>coap_msg_set_token(coap_msg_t *msg, char *buf, size_t len)</code>	86
4.10.3.22	<code>coap_msg_set_type(coap_msg_t *msg, unsigned type)</code>	87
4.10.3.23	<code>coap_msg_uri_path_to_str(coap_msg_t *msg, char *buf, size_t len)</code>	87
4.11	<code>lib/src/coap_server.c</code> File Reference	88
4.11.1	Detailed Description	89
4.11.2	Macro Definition Documentation	89
4.11.2.1	<code>COAP_SERVER_ACK_TIMEOUT_SEC</code>	89
4.11.2.2	<code>COAP_SERVER_DTLS_HANDSHAKE_ATTEMPTS</code>	89
4.11.2.3	<code>COAP_SERVER_DTLS_MTU</code>	89
4.11.2.4	<code>COAP_SERVER_DTLS_NUM_DH_BITS</code>	89
4.11.2.5	<code>COAP_SERVER_DTLS_PRIORITIES</code>	89
4.11.2.6	<code>COAP_SERVER_DTLS_RETRANS_TIMEOUT</code>	90
4.11.2.7	<code>COAP_SERVER_DTLS_TOTAL_TIMEOUT</code>	90
4.11.2.8	<code>COAP_SERVER_MAX_RETRANSMIT</code>	90
4.11.3	Function Documentation	90
4.11.3.1	<code>coap_server_add_sep_resp_uri_path(coap_server_t *server, const char *str)</code>	90
4.11.3.2	<code>coap_server_create(coap_server_t *server, coap_server_trans_handler_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)</code>	90
4.11.3.3	<code>coap_server_destroy(coap_server_t *server)</code>	91
4.11.3.4	<code>coap_server_get_next_msg_id(coap_server_t *server)</code>	91
4.11.3.5	<code>coap_server_run(coap_server_t *server)</code>	91
4.11.3.6	<code>coap_server_trans_handle_blockwise(coap_server_trans_t *trans, coap_msg_t *req, coap_msg_t *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, coap_server_trans_handler_t block_rx)</code>	92
	Index	93

Chapter 1

Class Index

1.1 Class List

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

coap_client_t		
Client structure	5
coap_mem_t		
Memory allocator structure	7
coap_msg_op		
Option structure	8
coap_msg_op_list_t		
Option linked-list structure	9
coap_msg_t		
Message structure	10
coap_server		
Server structure	12
coap_server_path		
URI path structure	14
coap_server_path_list_t		
URI path list structure	15
coap_server_trans		
Transaction structure	16

Chapter 2

File Index

2.1 File List

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

lib/include/ coap_client.h	
Include file for the FreeCoAP client library	21
lib/include/ coap_ipv.h	
Include file for the FreeCoAP IP Version (IPv4/IPv6) abstraction layer	24
lib/include/ coap_log.h	
Include file for the FreeCoAP logging module	26
lib/include/ coap_mem.h	
Include file for the FreeCoAP memory allocator	29
lib/include/ coap_msg.h	
Include file for the FreeCoAP message parser/formatter library	38
lib/include/ coap_server.h	
Include file for the FreeCoAP server library	56
lib/src/ coap_client.c	
Source file for the FreeCoAP client library	62
lib/src/ coap_log.c	
Source file for the FreeCoAP logging module	66
lib/src/ coap_mem.c	
Source file for the FreeCoAP memory allocator	68
lib/src/ coap_msg.c	
Source file for the FreeCoAP message parser/formatter library	77
lib/src/ coap_server.c	
Source file for the FreeCoAP server library	88

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_mem_t Struct Reference

Memory allocator structure.

```
#include <coap_mem.h>
```

Public Attributes

- char * [buf](#)
- size_t [num](#)
- size_t [len](#)
- char * [active](#)

3.2.1 Detailed Description

Memory allocator structure.

3.2.2 Member Data Documentation

3.2.2.1 char* coap_mem_t::active

Bitset marking active buffers

3.2.2.2 char* coap_mem_t::buf

Pointer to an array of buffers

3.2.2.3 size_t coap_mem_t::len

Length of each buffer

3.2.2.4 size_t coap_mem_t::num

Number of buffers

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

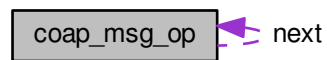
- lib/include/[coap_mem.h](#)

3.3 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](#)

3.3.1 Detailed Description

Option structure.

3.3.2 Member Data Documentation

3.3.2.1 unsigned coap_msg_op::len

Option length

3.3.2.2 struct coap_msg_op* coap_msg_op::next

Pointer to the next option structure in the list

3.3.2.3 unsigned coap_msg_op::num

Option number

3.3.2.4 char* coap_msg_op::val

Pointer to a buffer containing the option value

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

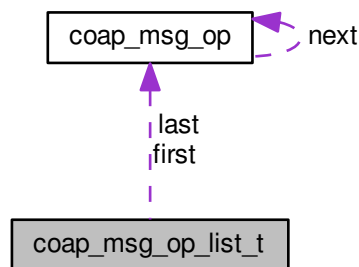
- lib/include/coap_msg.h

3.4 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](#)

3.4.1 Detailed Description

Option linked-list structure.

3.4.2 Member Data Documentation

3.4.2.1 coap_msg_op_t* coap_msg_op_list_t::first

Pointer to the first option structure in the list

3.4.2.2 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:

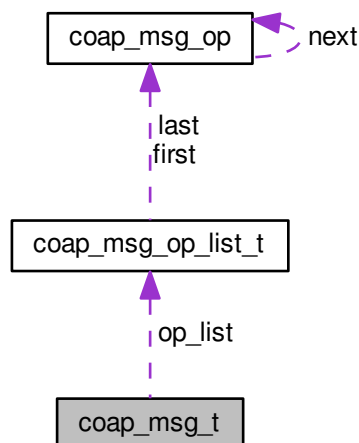
- lib/include/coap_msg.h

3.5 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`

3.5.1 Detailed Description

Message structure.

3.5.2 Member Data Documentation

3.5.2.1 unsigned coap_msg_t::code_class

Code class

3.5.2.2 unsigned coap_msg_t::code_detail

Code detail

3.5.2.3 unsigned coap_msg_t::msg_id

Message ID

3.5.2.4 coap_msg_op_list_t coap_msg_t::op_list

Option list

3.5.2.5 char* coap_msg_t::payload

Pointer to a buffer containing the payload

3.5.2.6 size_t coap_msg_t::payload_len

Length of the payload

3.5.2.7 char coap_msg_t::token[COAP_MSG_MAX_TOKEN_LEN]

Token value

3.5.2.8 unsigned coap_msg_t::token_len

Token length

3.5.2.9 coap_msg_type_t coap_msg_t::type

Message type

3.5.2.10 unsigned coap_msg_t::ver

CoAP version

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

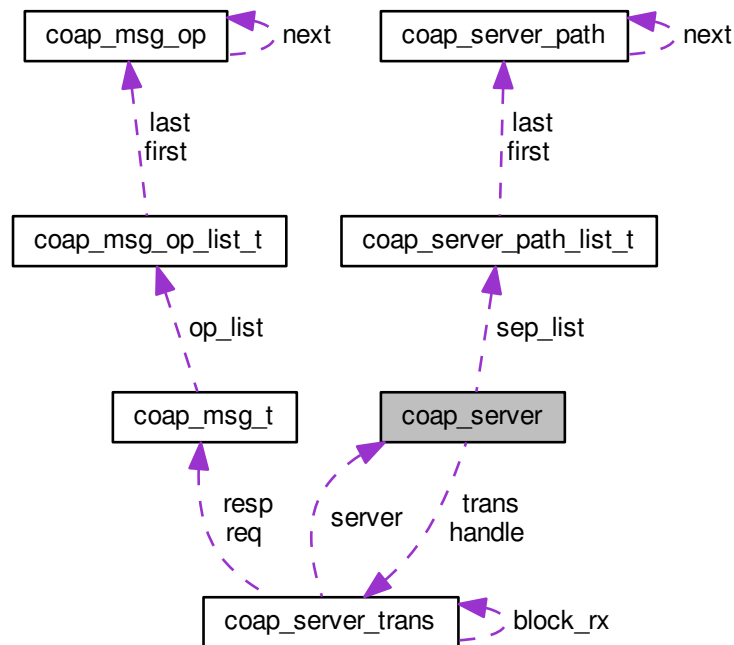
- lib/include/coap_msg.h

3.6 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`]
- `coap_server_trans_handler_t` `handle`
- `gnutls_certificate_credentials_t` `cred`
- `gnutls_priority_t` `priority`
- `gnutls_dh_params_t` `dh_params`

3.6.1 Detailed Description

Server structure.

3.6.2 Member Data Documentation

3.6.2.1 `gnutls_certificate_credentials_t coap_server::cred`

DTLS credentials

3.6.2.2 `gnutls_dh_params_t coap_server::dh_params`

Diffie-Hellman parameters

3.6.2.3 `coap_server_trans_handler_t coap_server::handle`

Call-back function to handle requests and generate responses

3.6.2.4 `unsigned coap_server::msg_id`

Last message ID value used in a response message

3.6.2.5 `gnutls_priority_t coap_server::priority`

DTLS priorities

3.6.2.6 `int coap_server::sd`

Socket descriptor

3.6.2.7 `coap_server_path_list_t coap_server::sep_list`

List of URI paths that require separate responses

3.6.2.8 `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:

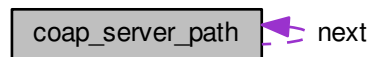
- `lib/include/coap_server.h`

3.7 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](#)

3.7.1 Detailed Description

URI path structure.

3.7.2 Member Data Documentation

3.7.2.1 struct [coap_server_path](#)* [coap_server_path::next](#)

Pointer to the next URI path structure in the list

3.7.2.2 char* [coap_server_path::str](#)

String containing a path

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

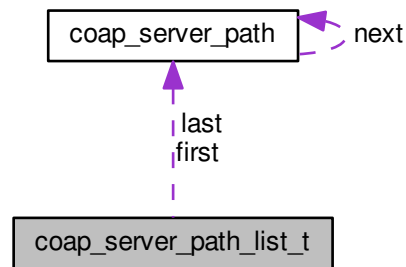
- lib/include/[coap_server.h](#)

3.8 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](#)

3.8.1 Detailed Description

URI path list structure.

3.8.2 Member Data Documentation

3.8.2.1 coap_server_path_t* coap_server_path_list_t::first

Pointer to the first URI path structure in the list

3.8.2.2 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:

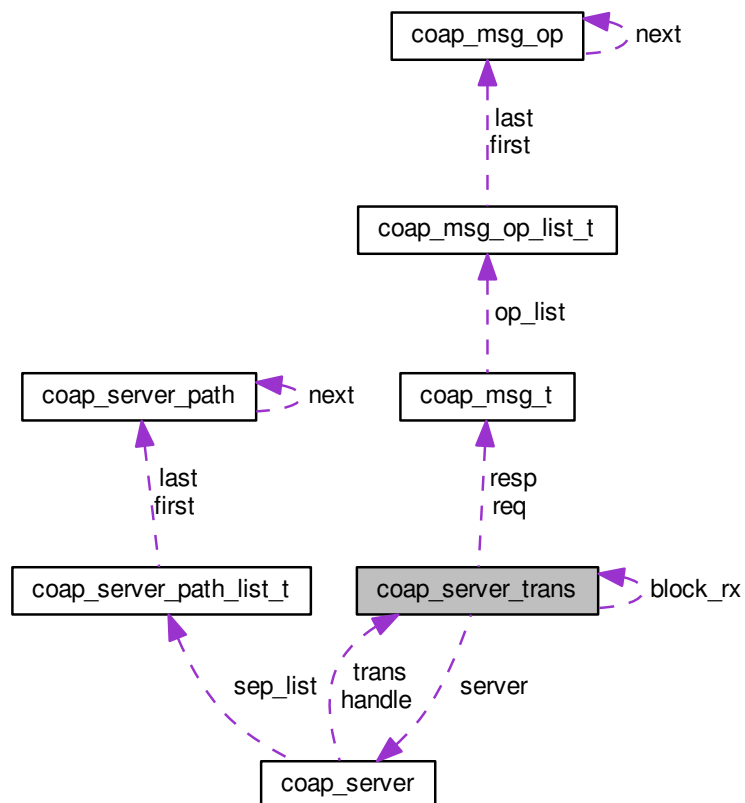
- lib/include/[coap_server.h](#)

3.9 coap_server_trans Struct Reference

Transaction structure.

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

Collaboration diagram for coap_server_trans:



Public Attributes

- int `active`
- `coap_server_trans_type_t` `type`
- 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`
- char * `body`

- `size_t` [body_len](#)
- `size_t` [body_end](#)
- `unsigned` [block1_size](#)
- `unsigned` [block2_size](#)
- `size_t` [block1_next](#)
- `size_t` [block2_next](#)
- `char` [block_uri](#) [COAP_MSG_OP_URI_PATH_MAX_LEN+1]
- `coap_msg_success_t` [block_detail](#)
- `coap_server_trans_handler_t` [block_rx](#)
- `struct` `coap_server` * [server](#)
- `gnutls_session_t` [session](#)

3.9.1 Detailed Description

Transaction structure.

3.9.2 Member Data Documentation

3.9.2.1 `int` `coap_server_trans::active`

Flag to indicate if this transaction structure contains valid data

3.9.2.2 `size_t` `coap_server_trans::block1_next`

Byte offset of the next block in the request

3.9.2.3 `unsigned` `coap_server_trans::block1_size`

Block1 size for blockwise transfers

3.9.2.4 `size_t` `coap_server_trans::block2_next`

Byte offset of the next block in the response

3.9.2.5 `unsigned` `coap_server_trans::block2_size`

Block2 size for blockwise transfers

3.9.2.6 `coap_msg_success_t` `coap_server_trans::block_detail`

Code detail for a PUT or POST blockwise operation

3.9.2.7 `coap_server_trans_handler_t` `coap_server_trans::block_rx`

User-supplied callback function to be called when the body of a blockwise transfer has been fully received

3.9.2.8 `char` `coap_server_trans::block_uri`[`COAP_MSG_OP_URI_PATH_MAX_LEN+1`]

The URI for the current blockwise transfer

3.9.2.9 `char*` `coap_server_trans::body`

Pointer to a buffer for blockwise transfers

3.9.2.10 `size_t` `coap_server_trans::body_end`

Amount of relevant data in the buffer for blockwise transfers

3.9.2.11 `size_t` `coap_server_trans::body_len`

Length of the buffer for blockwise transfers

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

String to hold the client address

3.9.2.13 `coap_ipv_sockaddr_in_t` `coap_server_trans::client_sin`

Socket structure

3.9.2.14 `socklen_t` `coap_server_trans::client_sin_len`

Socket structure length

3.9.2.15 `time_t` `coap_server_trans::last_use`

The time that this transaction structure was last used

3.9.2.16 `unsigned` `coap_server_trans::num_retrans`

Current number of retransmissions

3.9.2.17 `coap_msg_t coap_server_trans::req`

Last request message received for this transaction

3.9.2.18 `coap_msg_t coap_server_trans::resp`

Last response message sent for this transaction

3.9.2.19 `struct coap_server* coap_server_trans::server`

Pointer to the containing server structure

3.9.2.20 `gnutls_session_t coap_server_trans::session`

DTLS session

3.9.2.21 `struct timespec coap_server_trans::timeout`

Timeout value

3.9.2.22 `int coap_server_trans::timer_fd`

Timer file descriptor

3.9.2.23 `coap_server_trans_type_t coap_server_trans::type`

Transaction type

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

- `lib/include/coap_server.h`

Chapter 4

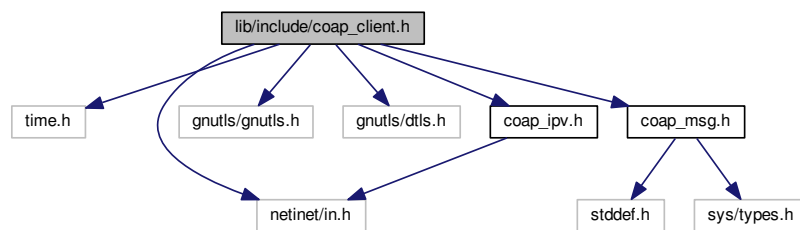
File Documentation

4.1 lib/include/coap_client.h File Reference

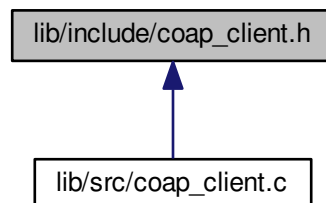
Include file for the FreeCoAP client library.

```
#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.
- ssize_t [coap_client_exchange_blockwise](#) ([coap_client_t](#) *client, [coap_msg_t](#) *req, [coap_msg_t](#) *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, int have_resp)
Exchange with the server using blockwise transfers.

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
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

4.1.3.2 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

4.1.3.4 `ssize_t coap_client_exchange_blockwise (coap_client_t * client, coap_msg_t * req, coap_msg_t * resp, unsigned block1_size, unsigned block2_size, char * body, size_t body_len, int have_resp)`

Exchange with the server using blockwise transfers.

The calling application should not pass in a request message that contains a block1 or block2 option. This function will add block1 and block2 options internally. 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
in	<i>block1_size</i>	Block1 size
in	<i>block2_size</i>	Block2 size
in	<i>body</i>	Pointer to a buffer to hold the body
in	<i>body_len</i>	Length of the buffer to hold the body
in	<i>have_resp</i>	Flag to indicate that the first response has already been received

Returns

Operation status

Return values

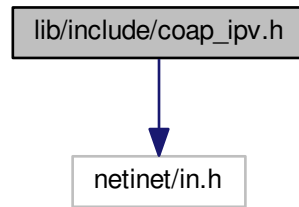
>=0	Length of the data received
<0	Error

4.2 lib/include/coap_ipv.h File Reference

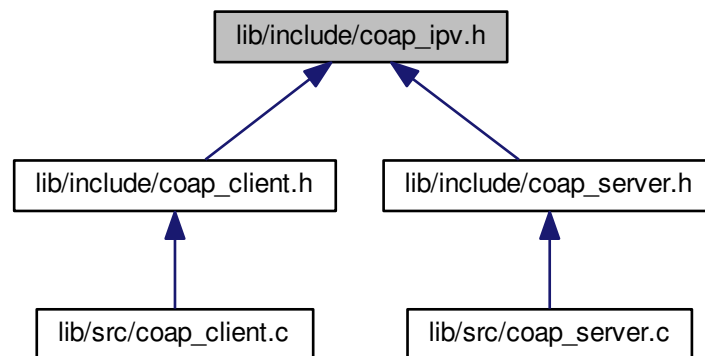
Include file for the FreeCoAP IP Version (IPv4/IPv6) abstraction layer.


```
#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_INET_ADDRSTRLEN INET_ADDRSTRLEN`
- `#define COAP_IPV_SIN_ADDR sin_addr`
- `#define COAP_IPV_SIN_PORT sin_port`

Typedefs

- `typedef struct sockaddr_in coap_ipv_sockaddr_in_t`

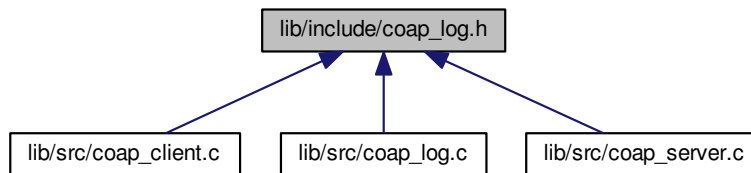
4.2.1 Detailed Description

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

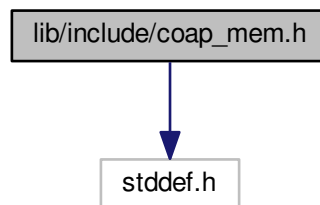
in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

4.4 lib/include/coap_mem.h File Reference

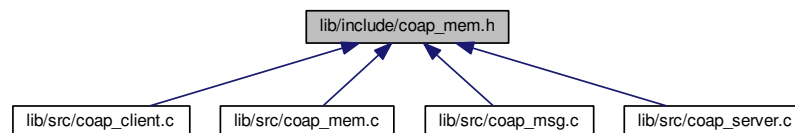
Include file for the FreeCoAP memory allocator.

```
#include <stddef.h>
```

Include dependency graph for coap_mem.h:



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



Classes

- struct [coap_mem_t](#)
Memory allocator structure.

Macros

- #define [coap_mem_get_buf](#)(mem) ((mem)->buf)
- #define [coap_mem_get_num](#)(mem) ((mem)->num)
- #define [coap_mem_get_len](#)(mem) ((mem)->len)
- #define [coap_mem_get_active](#)(mem) ((mem)->active)
- #define [coap_mem_get_active_len](#)(mem) ((mem)->num >> 3)

Functions

- int `coap_mem_create` (`coap_mem_t` *mem, size_t num, size_t len)
Initialise a memory allocator structure.
- void `coap_mem_destroy` (`coap_mem_t` *mem)
Deinitialise a memory allocator structure.
- void * `coap_mem_alloc` (`coap_mem_t` *mem, size_t len)
Allocate a buffer from a memory allocator.
- void `coap_mem_free` (`coap_mem_t` *mem, void *buf)
Return a buffer back to a memory allocator.
- int `coap_mem_small_create` (size_t num, size_t len)
Initialise the small memory allocator.
- void `coap_mem_small_destroy` (void)
Deinitialise the small memory allocator.
- char * `coap_mem_small_get_buf` (void)
Get the array of buffers in the small memory allocator.
- size_t `coap_mem_small_get_num` (void)
Get the number of buffers in the small memory allocator.
- size_t `coap_mem_small_get_len` (void)
Get the length of each buffer in the small memory allocator.
- size_t `coap_mem_small_get_active_len` (void)
Get the length of the active bitset from the small memory allocator.
- char * `coap_mem_small_get_active` (void)
Get the active bitset from the small memory allocator.
- void * `coap_mem_small_alloc` (size_t len)
Allocate a buffer from the small memory allocator.
- void `coap_mem_small_free` (void *buf)
Return a buffer back to the small memory allocator.
- int `coap_mem_medium_create` (size_t num, size_t len)
Initialise the medium memory allocator.
- void `coap_mem_medium_destroy` (void)
Deinitialise the medium memory allocator.
- char * `coap_mem_medium_get_buf` (void)
Get the array of buffers in the medium memory allocator.
- size_t `coap_mem_medium_get_num` (void)
Get the number of buffers in the medium memory allocator.
- size_t `coap_mem_medium_get_len` (void)
Get the length of each buffer in the medium memory allocator.
- size_t `coap_mem_medium_get_active_len` (void)
Get the length of the active bitset from the medium memory allocator.
- char * `coap_mem_medium_get_active` (void)
Get the active bitset from the medium memory allocator.
- void * `coap_mem_medium_alloc` (size_t len)
Allocate a buffer from the medium memory allocator.
- void `coap_mem_medium_free` (void *buf)
Return a buffer back to the medium memory allocator.
- int `coap_mem_large_create` (size_t num, size_t len)
Initialise the large memory allocator.
- void `coap_mem_large_destroy` (void)
Deinitialise the large memory allocator.
- char * `coap_mem_large_get_buf` (void)

- Get the array of buffers in the large memory allocator.*
- `size_t coap_mem_large_get_num` (void)
- Get the number of buffers in the large memory allocator.*
- `size_t coap_mem_large_get_len` (void)
- Get the length of each buffer in the large memory allocator.*
- `size_t coap_mem_large_get_active_len` (void)
- Get the length of the active bitset from the large memory allocator.*
- `char * coap_mem_large_get_active` (void)
- Get the active bitset from the large memory allocator.*
- `void * coap_mem_large_alloc` (size_t len)
- Allocate a buffer from the large memory allocator.*
- `void coap_mem_large_free` (void *buf)
- Return a buffer back to the large memory allocator.*
- `int coap_mem_all_create` (size_t small_num, size_t small_len, size_t medium_num, size_t medium_len, size_t large_num, size_t large_len)
- Initialise all memory allocators.*
- `void coap_mem_all_destroy` (void)
- Deinitialise all memory allocators.*

4.4.1 Detailed Description

Include file for the FreeCoAP memory allocator.

4.4.2 Macro Definition Documentation

4.4.2.1 `#define coap_mem_get_active(mem) ((mem)->active)`

Get the active bitset from a memory allocator

4.4.2.2 `#define coap_mem_get_active_len(mem) ((mem)->num >> 3)`

Get the length of the active bitset from a memory allocator

4.4.2.3 `#define coap_mem_get_buf(mem) ((mem)->buf)`

Get the array of buffers in a memory allocator

4.4.2.4 `#define coap_mem_get_len(mem) ((mem)->len)`

Get the length of each buffer in a memory allocator

4.4.2.5 `#define coap_mem_get_num(mem) ((mem)->num)`

Get the number of buffers in a memory allocator

4.4.3 Function Documentation

4.4.3.1 `int coap_mem_all_create (size_t small_num, size_t small_len, size_t medium_num, size_t medium_len, size_t large_num, size_t large_len)`

Initialise all memory allocators.

Parameters

in	<i>small_num</i>	Number of small buffers
in	<i>small_len</i>	Length of each small buffer
in	<i>medium_num</i>	Number of medium buffers
in	<i>medium_len</i>	Length of each medium buffer
in	<i>large_num</i>	Number of large buffers
in	<i>large_len</i>	Length of each large buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.4.3.2 void* coap_mem_alloc (coap_mem_t * mem, size_t len)

Allocate a buffer from a memory allocator.

Parameters

in, out	<i>mem</i>	Pointer to a memory allocator
in	<i>len</i>	Length of the buffer

Returns

Pointer to a buffer or NULL

4.4.3.3 int coap_mem_create (coap_mem_t * mem, size_t num, size_t len)

Initialise a memory allocator structure.

Parameters

out	<i>mem</i>	Pointer to a memory allocator
in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.4.3.4 void coap_mem_destroy (coap_mem_t * mem)

Deinitialise a memory allocator structure.

Parameters

in, out	mem	Pointer to a memory allocator
---------	-----	-------------------------------

4.4.3.5 void coap_mem_free (coap_mem_t * mem, void * buf)

Return a buffer back to a memory allocator.

Parameters

in, out	mem	Pointer to a memory allocator
in	buf	Pointer to a buffer

4.4.3.6 void* coap_mem_large_alloc (size_t len)

Allocate a buffer from the large memory allocator.

Parameters

in	len	Length of the buffer
----	-----	----------------------

Returns

Pointer to a buffer or NULL

4.4.3.7 int coap_mem_large_create (size_t num, size_t len)

Initialise the large memory allocator.

Parameters

in	num	Number of buffers
in	len	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.4.3.8 void coap_mem_large_free (void * *buf*)

Return a buffer back to the large memory allocator.

Parameters

in	<i>buf</i>	Pointer to a buffer
----	------------	---------------------

4.4.3.9 char* coap_mem_large_get_active (void)

Get the active bitset from the large memory allocator.

Returns

the active bitset from the large memory allocator

4.4.3.10 size_t coap_mem_large_get_active_len (void)

Get the length of the active bitset from the large memory allocator.

Returns

Length of the active bitset from the large memory allocator

4.4.3.11 char* coap_mem_large_get_buf (void)

Get the array of buffers in the large memory allocator.

Returns

Pointer to the array of buffers

4.4.3.12 `size_t coap_mem_large_get_len (void)`

Get the length of each buffer in the large memory allocator.

Returns

Length of each buffer

4.4.3.13 `size_t coap_mem_large_get_num (void)`

Get the number of buffers in the large memory allocator.

Returns

Number of buffers

4.4.3.14 `void* coap_mem_medium_alloc (size_t len)`

Allocate a buffer from the medium memory allocator.

Parameters

in	<i>len</i>	Length of the buffer
----	------------	----------------------

Returns

Pointer to a buffer or NULL

4.4.3.15 `int coap_mem_medium_create (size_t num, size_t len)`

Initialise the medium memory allocator.

Parameters

in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.4.3.16 void coap_mem_medium_free (void * *buf*)

Return a buffer back to the medium memory allocator.

Parameters

in	<i>buf</i>	Pointer to a buffer
----	------------	---------------------

4.4.3.17 char* coap_mem_medium_get_active (void)

Get the active bitset from the medium memory allocator.

Returns

the active bitset from the medium memory allocator

4.4.3.18 size_t coap_mem_medium_get_active_len (void)

Get the length of the active bitset from the medium memory allocator.

Returns

Length of the active bitset from the medium memory allocator

4.4.3.19 char* coap_mem_medium_get_buf (void)

Get the array of buffers in the medium memory allocator.

Returns

Pointer to the array of buffers

4.4.3.20 size_t coap_mem_medium_get_len (void)

Get the length of each buffer in the medium memory allocator.

Returns

Length of each buffer

4.4.3.21 size_t coap_mem_medium_get_num (void)

Get the number of buffers in the medium memory allocator.

Returns

Number of buffers

4.4.3.22 void* coap_mem_small_alloc (size_t *len*)

Allocate a buffer from the small memory allocator.

Parameters

in	<i>len</i>	Length of the buffer
----	------------	----------------------

Returns

Pointer to a buffer or NULL

4.4.3.23 int coap_mem_small_create (size_t *num*, size_t *len*)

Initialise the small memory allocator.

Parameters

in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.4.3.24 void coap_mem_small_free (void * *buf*)

Return a buffer back to the small memory allocator.

Parameters

in	<i>buf</i>	Pointer to a buffer
----	------------	---------------------

4.4.3.25 char* coap_mem_small_get_active (void)

Get the active bitset from the small memory allocator.

Returns

the active bitset from the small memory allocator

4.4.3.26 `size_t coap_mem_small_get_active_len (void)`

Get the length of the active bitset from the small memory allocator.

Returns

Length of the active bitset from the small memory allocator

4.4.3.27 `char* coap_mem_small_get_buf (void)`

Get the array of buffers in the small memory allocator.

Returns

Pointer to the array of buffers

4.4.3.28 `size_t coap_mem_small_get_len (void)`

Get the length of each buffer in the small memory allocator.

Returns

Length of each buffer

4.4.3.29 `size_t coap_mem_small_get_num (void)`

Get the number of buffers in the small memory allocator.

Returns

Number of buffers

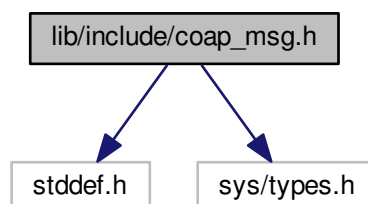
4.5 `lib/include/coap_msg.h` File Reference

Include file for the FreeCoAP message parser/formatter library.

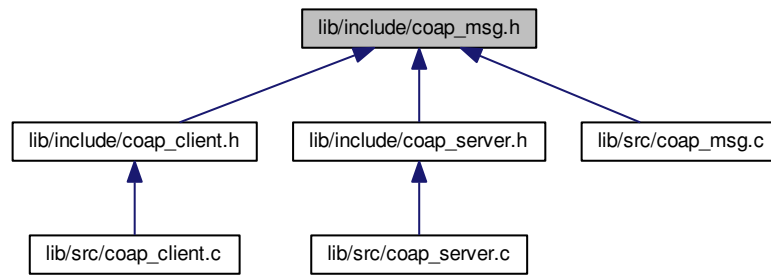
```
#include <stddef.h>
```

```
#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_MAX_PAYLOAD_LEN](#) 1024
- #define [coap_msg_block_szx_to_size](#)(szx) (1 << ((szx) + 4))
- #define [coap_msg_block_start_to_num](#)(start, szx) ((start) >> ((szx) + 4))
- #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_CHANGED` = 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_FMT` = 15 }

Client error response code detail enumeration.

- enum `coap_msg_server_err_t` { `COAP_MSG_INT_SERVER_ERR` = 0, `COAP_MSG_NOT_IMPL` = 1, `COAP_MSG_BAD_GATEWAY` = 2, `COAP_MSG_SERV_UNAVAIL` = 3, `COAP_MSG_GATEWAY_TIMEOUT` = 4, `COAP_MSG_PROXY_NOT_SUP` = 5 }

Server error response code detail enumeration.

- enum `coap_msg_op_num_t` { `COAP_MSG_IF_MATCH` = 1, `COAP_MSG_URI_HOST` = 3, `COAP_MSG_ETAG` = 4, `COAP_MSG_IF_NONE_MATCH` = 5, `COAP_MSG_URI_PORT` = 7, `COAP_MSG_LOCATION_PATH` = 8, `COAP_MSG_URI_PATH` = 11, `COAP_MSG_CONTENT_FORMAT` = 12, `COAP_MSG_MAX_AGE` = 14, `COAP_MSG_URI_QUERY` = 15, `COAP_MSG_ACCEPT` = 17, `COAP_MSG_LOCATION_QUERY` = 20, `COAP_MSG_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_calc_block_szx](#) (unsigned size)
Calculate block size exponent from block size.
- 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.
- int [coap_msg_parse_block_op](#) (unsigned *num, unsigned *more, unsigned *size, coap_msg_t *msg, int type)
Find and parse a Block1 or Block2 option in 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 an option to a message structure.
- int [coap_msg_set_payload](#) (coap_msg_t *msg, char *buf, size_t len)
Set the payload in a message.
- void [coap_msg_clear_payload](#) (coap_msg_t *msg)
Clear 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.

4.5.1 Detailed Description

Include file for the FreeCoAP message parser/formatter library.

4.5.2 Macro Definition Documentation

4.5.2.1 `#define coap_msg_block_start_to_num(start, szx) ((start) >> ((szx) + 4))`

Convert a start byte value to a block num value

4.5.2.2 `#define coap_msg_block_szx_to_size(szx) (1 << ((szx) + 4))`

Convert a block size exponent value to a size value

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

Get the code class from a message

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

Get the code detail from a message

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

Get the first option from a message

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

Get the message ID from message

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

Get the payload from a message

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

Get the payload length from a message

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

Get the token from a message

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

Get the token length from a message

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

Get the type from a message

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

Get the version from a message

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

Indicate if a message is empty

4.5.2.14 `#define COAP_MSG_MAX_BUF_LEN 1152`

Maximum buffer length for header and payload

4.5.2.15 `#define COAP_MSG_MAX_CODE_CLASS 7`

Maximum code class

4.5.2.16 `#define COAP_MSG_MAX_CODE_DETAIL 31`

Maximum code detail

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

Maximum message ID

4.5.2.18 `#define COAP_MSG_MAX_PAYLOAD_LEN 1024`

Maximum buffer length for payload

4.5.2.19 `#define COAP_MSG_MAX_TOKEN_LEN 8`

Maximum token length

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

Get the option length from an option

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

Get the next pointer from an option

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

Get the option number from an option

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

Get the option value from an option

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

Maximum block size for a Block1 or Block2 option

4.5.2.25 `#define COAP_MSG_OP_MAX_BLOCK_VAL_LEN 3`

Maximum buffer length for a Block1 or Block2 option value

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

Indicate if an option is critical

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

Indicate if an option is unsafe to forward

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

Indicate if an option is not part of the cache key

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

Set the option length in an option

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

Set the next pointer in an option

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

Set the option number in an option

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

Set the option value in an option

4.5.2.33 `#define COAP_MSG_OP_URI_PATH_MAX_LEN 256`

Maximum buffer length for a reconstructed URI path

4.5.2.34 `#define COAP_MSG_VER 0x01`

CoAP version

4.5.3 Enumeration Type Documentation

4.5.3.1 `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

4.5.3.2 `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

4.5.3.3 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

4.5.3.4 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

4.5.3.5 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

4.5.3.6 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

4.5.3.7 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

4.5.4 Function Documentation

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

Add an option 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

4.5.4.2 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

4.5.4.3 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

4.5.4.4 void coap_msg_clear_payload (coap_msg_t * msg)

Clear the payload in a message.

Free the buffer in the message structure containing the current payload if there is one.

Parameters

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

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

Copy a message.

Parameters

in, 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
<0	Error

4.5.4.6 void coap_msg_create (coap_msg_t * *msg*)

Initialise a message structure.

Parameters

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

4.5.4.7 void coap_msg_destroy (coap_msg_t * *msg*)

Deinitialise a message structure.

Parameters

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

4.5.4.8 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

4.5.4.9 void coap_msg_gen_rand_str (char * *buf*, size_t *len*)

Generate a random string of bytes.

Parameters

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

4.5.4.10 int coap_msg_op_calc_block_szx (unsigned *size*)

Calculate block size exponent from block size.

Parameters

in	<i>size</i>	Block size
----	-------------	------------

Returns

Block size exponent or error code

Return values

≥ 0	Block size exponent
< 0	Error

4.5.4.11 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 a buffer to store the option value
in	<i>len</i>	Length of the buffer
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

4.5.4.12 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

4.5.4.13 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

4.5.4.14 `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

4.5.4.15 `int coap_msg_parse_block_op (unsigned * num, unsigned * more, unsigned * size, coap_msg_t * msg, int type)`

Find and parse a Block1 or Block2 option in a message.

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 bytes)
in	<i>msg</i>	Pointer to a message
in	<i>type</i>	Block option type: COAP_MSG_BLOCK1 or COAP_MSG_BLOCK2

Returns

Operation status

Return values

1	Block option not found
0	Success
<0	Error

4.5.4.16 `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</i> ↔ <i>_id</i>	Pointer to a field to store the message ID value

Returns

Operation status

Return values

0	Success
<0	Error

4.5.4.17 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
---------	------------	--------------------------------

4.5.4.18 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

4.5.4.19 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

4.5.4.20 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

4.5.4.21 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

4.5.4.22 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

4.5.4.23 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

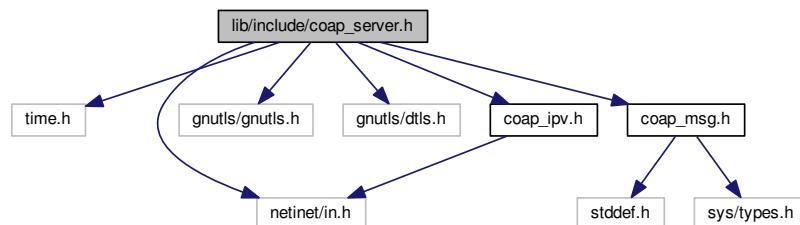
The number of bytes that would be written to the buffer it was large enough

4.6 lib/include/coap_server.h File Reference

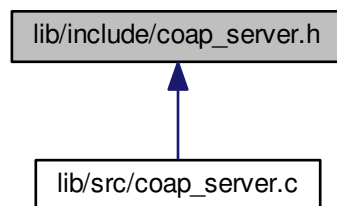
Include file for the FreeCoAP server library.

```
#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
- #define `coap_server_trans_get_type`(trans) ((trans)->type)
- #define `coap_server_trans_get_req`(trans) (&(trans)->req)
- #define `coap_server_trans_get_resp`(trans) (&(trans)->resp)
- #define `coap_server_trans_get_body`(trans) ((trans)->body)
- #define `coap_server_trans_get_body_len`(trans) ((trans)->body_len)
- #define `coap_server_trans_get_body_end`(trans) ((trans)->body_end)
- #define `coap_server_trans_set_body_end`(trans, i) ((trans)->body_end = (i))

Typedefs

- typedef int(* `coap_server_trans_handler_t`) (struct `coap_server_trans` *trans, `coap_msg_t` *req, `coap_msg_t` *resp)
Server transaction handler callback function.
- 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_trans_type_t` {
 `COAP_SERVER_TRANS_REGULAR` = 0, `COAP_SERVER_TRANS_BLOCKWISE_GET` = 1, `COAP_SERVER_TRANS_BLOCKWISE_PUT1` = 2, `COAP_SERVER_TRANS_BLOCKWISE_PUT2` = 3,
 `COAP_SERVER_TRANS_BLOCKWISE_POST1` = 4, `COAP_SERVER_TRANS_BLOCKWISE_POST2` = 5
}
Transaction type enumeration.
- enum `coap_server_resp_t` { `COAP_SERVER_PIGGYBACKED` = 0, `COAP_SERVER_SEPARATE` = 1 }
Response type enumeration.

Functions

- int `coap_server_trans_handle_blockwise` (`coap_server_trans_t` *trans, `coap_msg_t` *req, `coap_msg_t` *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, `coap_server_trans_handler_t` block_rx)
Handle a library-level blockwise transfer.
- int `coap_server_create` (`coap_server_t` *server, `coap_server_trans_handler_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.
- 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.

4.6.1 Detailed Description

Include file for the FreeCoAP server library.

4.6.2 Macro Definition Documentation

4.6.2.1 `#define COAP_SERVER_ADDR_BUF_LEN 128`

Buffer length for host addresses

4.6.2.2 `#define COAP_SERVER_DIAG_PAYLOAD_LEN 128`

Buffer length for diagnostic payloads

4.6.2.3 `#define COAP_SERVER_NUM_TRANS 8`

Maximum number of active transactions per server

4.6.2.4 `#define coap_server_trans_get_body(trans) ((trans)->body)`

Get the body of a blockwise transfer

4.6.2.5 `#define coap_server_trans_get_body_end(trans) ((trans)->body_end)`

Get the amount of relevant data in body of a blockwise transfer

4.6.2.6 `#define coap_server_trans_get_body_len(trans) ((trans)->body_len)`

Get the length of the body of a blockwise transfer

4.6.2.7 `#define coap_server_trans_get_req(trans) (&(trans)->req)`

Get the last request message received for this transaction

4.6.2.8 `#define coap_server_trans_get_resp(trans) (&(trans)->resp)`

Get the last response message sent for this transaction

4.6.2.9 `#define coap_server_trans_get_type(trans) ((trans)->type)`

Get the type of transaction

4.6.2.10 `#define coap_server_trans_set_body_end(trans, i) ((trans)->body_end = (i))`

Get the amount of relevant data in body of a blockwise transfer

4.6.3 Typedef Documentation

4.6.3.1 `typedef int(* coap_server_trans_handler_t)(struct coap_server_trans *trans, coap_msg_t *req, coap_msg_t *resp)`

Server transaction handler callback function.

Parameters

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

Returns

Operation status

Return values

0	Success
<0	Error

4.6.4 Enumeration Type Documentation

4.6.4.1 enum coap_server_resp_t

Response type enumeration.

Enumerator

COAP_SERVER_PIGGYBACKED Piggybacked response

COAP_SERVER_SEPARATE Separate response

4.6.4.2 enum coap_server_trans_type_t

Transaction type enumeration.

Enumerator

COAP_SERVER_TRANS_REGULAR Regular (i.e. non-blockwise) transaction

COAP_SERVER_TRANS_BLOCKWISE_GET Blockwise GET transaction

COAP_SERVER_TRANS_BLOCKWISE_PUT1 Request phase of a blockwise PUT transaction

COAP_SERVER_TRANS_BLOCKWISE_PUT2 Response phase of a blockwise PUT transaction

COAP_SERVER_TRANS_BLOCKWISE_POST1 Request phase of a Blockwise POST transaction

COAP_SERVER_TRANS_BLOCKWISE_POST2 Response phase of a Blockwise POST transaction

4.6.5 Function Documentation

4.6.5.1 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

4.6.5.2 `int coap_server_create (coap_server_t * server, coap_server_trans_handler_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>	String containing the host address of the server
in	<i>port</i>	String containing the 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

4.6.5.3 `void coap_server_destroy (coap_server_t * server)`

Deinitialise a server structure.

Parameters

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

4.6.5.4 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

4.6.5.5 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

4.6.5.6 int coap_server_trans_handle_blockwise (coap_server_trans_t * trans, coap_msg_t * req, coap_msg_t * resp, unsigned block1_size, unsigned block2_size, char * body, size_t body_len, coap_server_trans_handler_t block_rx)

Handle a library-level blockwise transfer.

Configure the transaction structure to do a library-level blockwise transfer. This function should be called by the application from the handle callback function.

Parameters

in, out	trans	Pointer to a transaction structure
in	req	Pointer to the request message
out	resp	Pointer to the response message
in	block1_size	Preferred block1 size
in	block2_size	Preferred block2 size

- `#define COAP_CLIENT_DTLS_MTU COAP_MSG_MAX_BUF_LEN`
- `#define COAP_CLIENT_DTLS_RETRANS_TIMEOUT 1000`
- `#define COAP_CLIENT_DTLS_TOTAL_TIMEOUT 60000`
- `#define COAP_CLIENT_DTLS_HANDSHAKE_ATTEMPTS 60`
- `#define COAP_CLIENT_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDENCE"`

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.
- `ssize_t coap_client_exchange_blockwise (coap_client_t *client, coap_msg_t *req, coap_msg_t *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, int have_resp)`
Exchange with the server using blockwise transfers.

4.7.1 Detailed Description

Source file for the FreeCoAP client library.

4.7.2 Macro Definition Documentation

4.7.2.1 `#define COAP_CLIENT_ACK_TIMEOUT_SEC 2`

Minimum delay to wait before retransmitting a confirmable message

4.7.2.2 `#define COAP_CLIENT_DTLS_HANDSHAKE_ATTEMPTS 60`

Maximum number of DTLS handshake attempts

4.7.2.3 `#define COAP_CLIENT_DTLS_MTU COAP_MSG_MAX_BUF_LEN`

Maximum transmission unit excluding the UDP and IPv6 headers

4.7.2.4 `#define COAP_CLIENT_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDENCE"`

DTLS priorities

4.7.2.5 `#define COAP_CLIENT_DTLS_RETRANS_TIMEOUT 1000`

Retransmission timeout (msec) for the DTLS handshake

4.7.2.6 `#define COAP_CLIENT_DTLS_TOTAL_TIMEOUT 60000`

Total timeout (msec) for the DTLS handshake

4.7.2.7 `#define COAP_CLIENT_MAX_RETRANSMIT 4`

Maximum number of times a confirmable message can be retransmitted

4.7.2.8 `#define COAP_CLIENT_RESP_TIMEOUT_SEC 30`

Maximum amount of time to wait for a response

4.7.3 Function Documentation

4.7.3.1 `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

4.7.3.2 void coap_client_destroy (coap_client_t * client)

Deinitialise a client structure.

Parameters

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

4.7.3.3 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

4.7.3.4 ssize_t coap_client_exchange_blockwise (coap_client_t * client, coap_msg_t * req, coap_msg_t * resp, unsigned block1_size, unsigned block2_size, char * body, size_t body_len, int have_resp)

Exchange with the server using blockwise transfers.

The calling application should not pass in a request message that contains a block1 or block2 option. This function will add block1 and block2 options internally. 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
in	<i>block1_size</i>	Block1 size
in	<i>block2_size</i>	Block2 size
in	<i>body</i>	Pointer to a buffer to hold the body
in	<i>body_len</i>	Length of the buffer to hold the body
in	<i>have_resp</i>	Flag to indicate that the first response has already been received

Returns

Operation status

Return values

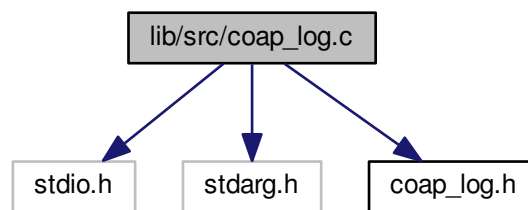
≥ 0	Length of the data received
< 0	Error

4.8 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.

4.8.1 Detailed Description

Source file for the FreeCoAP logging module.

4.8.2 Function Documentation

4.8.2.1 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

4.8.2.2 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

4.8.2.3 coap_log_level_t coap_log_get_level (void)

Get the log level.

Returns

The current log level

4.8.2.4 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

4.8.2.5 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

4.8.2.6 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

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

Log a warning message.

Parameters

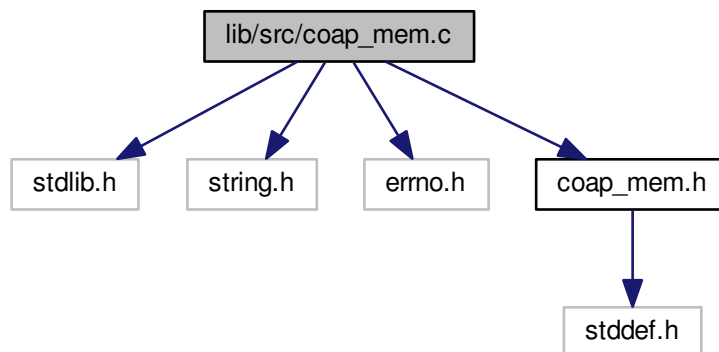
in	<i>msg</i>	String containing format specifiers
in	...	arguments for the format specifiers

4.9 lib/src/coap_mem.c File Reference

Source file for the FreeCoAP memory allocator.

```
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include "coap_mem.h"
```

Include dependency graph for coap_mem.c:



Functions

- int [coap_mem_create](#) ([coap_mem_t](#) *mem, size_t num, size_t len)
Initialise a memory allocator structure.
- void [coap_mem_destroy](#) ([coap_mem_t](#) *mem)
Deinitialise a memory allocator structure.
- void * [coap_mem_alloc](#) ([coap_mem_t](#) *mem, size_t len)
Allocate a buffer from a memory allocator.
- void [coap_mem_free](#) ([coap_mem_t](#) *mem, void *buf)
Return a buffer back to a memory allocator.
- int [coap_mem_small_create](#) (size_t num, size_t len)
Initialise the small memory allocator.
- void [coap_mem_small_destroy](#) (void)
Deinitialise the small memory allocator.
- char * [coap_mem_small_get_buf](#) (void)
Get the array of buffers in the small memory allocator.
- size_t [coap_mem_small_get_num](#) (void)
Get the number of buffers in the small memory allocator.
- size_t [coap_mem_small_get_len](#) (void)
Get the length of each buffer in the small memory allocator.
- size_t [coap_mem_small_get_active_len](#) (void)
Get the length of the active bitset from the small memory allocator.
- char * [coap_mem_small_get_active](#) (void)
Get the active bitset from the small memory allocator.
- void * [coap_mem_small_alloc](#) (size_t len)
Allocate a buffer from the small memory allocator.
- void [coap_mem_small_free](#) (void *buf)
Return a buffer back to the small memory allocator.
- int [coap_mem_medium_create](#) (size_t num, size_t len)
Initialise the medium memory allocator.
- void [coap_mem_medium_destroy](#) (void)

- Deinitialise the medium memory allocator.*
- char * `coap_mem_medium_get_buf` (void)
Get the array of buffers in the medium memory allocator.
- size_t `coap_mem_medium_get_num` (void)
Get the number of buffers in the medium memory allocator.
- size_t `coap_mem_medium_get_len` (void)
Get the length of each buffer in the medium memory allocator.
- size_t `coap_mem_medium_get_active_len` (void)
Get the length of the active bitset from the medium memory allocator.
- char * `coap_mem_medium_get_active` (void)
Get the active bitset from the medium memory allocator.
- void * `coap_mem_medium_alloc` (size_t len)
Allocate a buffer from the medium memory allocator.
- void `coap_mem_medium_free` (void *buf)
Return a buffer back to the medium memory allocator.
- int `coap_mem_large_create` (size_t num, size_t len)
Initialise the large memory allocator.
- void `coap_mem_large_destroy` (void)
Deinitialise the large memory allocator.
- char * `coap_mem_large_get_buf` (void)
Get the array of buffers in the large memory allocator.
- size_t `coap_mem_large_get_num` (void)
Get the number of buffers in the large memory allocator.
- size_t `coap_mem_large_get_len` (void)
Get the length of each buffer in the large memory allocator.
- size_t `coap_mem_large_get_active_len` (void)
Get the length of the active bitset from the large memory allocator.
- char * `coap_mem_large_get_active` (void)
Get the active bitset from the large memory allocator.
- void * `coap_mem_large_alloc` (size_t len)
Allocate a buffer from the large memory allocator.
- void `coap_mem_large_free` (void *buf)
Return a buffer back to the large memory allocator.
- int `coap_mem_all_create` (size_t small_num, size_t small_len, size_t medium_num, size_t medium_len, size_t large_num, size_t large_len)
Initialise all memory allocators.
- void `coap_mem_all_destroy` (void)
Deinitialise all memory allocators.

4.9.1 Detailed Description

Source file for the FreeCoAP memory allocator.

4.9.2 Function Documentation

4.9.2.1 int `coap_mem_all_create` (size_t *small_num*, size_t *small_len*, size_t *medium_num*, size_t *medium_len*, size_t *large_num*, size_t *large_len*)

Initialise all memory allocators.

Parameters

in	<i>small_num</i>	Number of small buffers
in	<i>small_len</i>	Length of each small buffer
in	<i>medium_num</i>	Number of medium buffers
in	<i>medium_len</i>	Length of each medium buffer
in	<i>large_num</i>	Number of large buffers
in	<i>large_len</i>	Length of each large buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.9.2.2 void* coap_mem_alloc (coap_mem_t * mem, size_t len)

Allocate a buffer from a memory allocator.

Parameters

in, out	<i>mem</i>	Pointer to a memory allocator
in	<i>len</i>	Length of the buffer

Returns

Pointer to a buffer or NULL

4.9.2.3 int coap_mem_create (coap_mem_t * mem, size_t num, size_t len)

Initialise a memory allocator structure.

Parameters

out	<i>mem</i>	Pointer to a memory allocator
in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.9.2.4 void coap_mem_destroy (coap_mem_t * mem)

Deinitialise a memory allocator structure.

Parameters

in, out	mem	Pointer to a memory allocator
---------	-----	-------------------------------

4.9.2.5 void coap_mem_free (coap_mem_t * mem, void * buf)

Return a buffer back to a memory allocator.

Parameters

in, out	mem	Pointer to a memory allocator
in	buf	Pointer to a buffer

4.9.2.6 void* coap_mem_large_alloc (size_t len)

Allocate a buffer from the large memory allocator.

Parameters

in	len	Length of the buffer
----	-----	----------------------

Returns

Pointer to a buffer or NULL

4.9.2.7 int coap_mem_large_create (size_t num, size_t len)

Initialise the large memory allocator.

Parameters

in	num	Number of buffers
in	len	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.9.2.8 void coap_mem_large_free (void * *buf*)

Return a buffer back to the large memory allocator.

Parameters

in	<i>buf</i>	Pointer to a buffer
----	------------	---------------------

4.9.2.9 char* coap_mem_large_get_active (void)

Get the active bitset from the large memory allocator.

Returns

the active bitset from the large memory allocator

4.9.2.10 size_t coap_mem_large_get_active_len (void)

Get the length of the active bitset from the large memory allocator.

Returns

Length of the active bitset from the large memory allocator

4.9.2.11 char* coap_mem_large_get_buf (void)

Get the array of buffers in the large memory allocator.

Returns

Pointer to the array of buffers

4.9.2.12 `size_t coap_mem_large_get_len (void)`

Get the length of each buffer in the large memory allocator.

Returns

Length of each buffer

4.9.2.13 `size_t coap_mem_large_get_num (void)`

Get the number of buffers in the large memory allocator.

Returns

Number of buffers

4.9.2.14 `void* coap_mem_medium_alloc (size_t len)`

Allocate a buffer from the medium memory allocator.

Parameters

in	<i>len</i>	Length of the buffer
----	------------	----------------------

Returns

Pointer to a buffer or NULL

4.9.2.15 `int coap_mem_medium_create (size_t num, size_t len)`

Initialise the medium memory allocator.

Parameters

in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.9.2.16 void coap_mem_medium_free (void * *buf*)

Return a buffer back to the medium memory allocator.

Parameters

<i>in</i>	<i>buf</i>	Pointer to a buffer
-----------	------------	---------------------

4.9.2.17 char* coap_mem_medium_get_active (void)

Get the active bitset from the medium memory allocator.

Returns

the active bitset from the medium memory allocator

4.9.2.18 size_t coap_mem_medium_get_active_len (void)

Get the length of the active bitset from the medium memory allocator.

Returns

Length of the active bitset from the medium memory allocator

4.9.2.19 char* coap_mem_medium_get_buf (void)

Get the array of buffers in the medium memory allocator.

Returns

Pointer to the array of buffers

4.9.2.20 size_t coap_mem_medium_get_len (void)

Get the length of each buffer in the medium memory allocator.

Returns

Length of each buffer

4.9.2.21 size_t coap_mem_medium_get_num (void)

Get the number of buffers in the medium memory allocator.

Returns

Number of buffers

4.9.2.22 void* coap_mem_small_alloc (size_t *len*)

Allocate a buffer from the small memory allocator.

Parameters

in	<i>len</i>	Length of the buffer
----	------------	----------------------

Returns

Pointer to a buffer or NULL

4.9.2.23 int coap_mem_small_create (size_t *num*, size_t *len*)

Initialise the small memory allocator.

Parameters

in	<i>num</i>	Number of buffers
in	<i>len</i>	Length of each buffer

Returns

Operation status

Return values

0	Success
<0	Error

4.9.2.24 void coap_mem_small_free (void * *buf*)

Return a buffer back to the small memory allocator.

Parameters

in	<i>buf</i>	Pointer to a buffer
----	------------	---------------------

4.9.2.25 char* coap_mem_small_get_active (void)

Get the active bitset from the small memory allocator.

Returns

the active bitset from the small memory allocator

4.9.2.26 `size_t coap_mem_small_get_active_len (void)`

Get the length of the active bitset from the small memory allocator.

Returns

Length of the active bitset from the small memory allocator

4.9.2.27 `char* coap_mem_small_get_buf (void)`

Get the array of buffers in the small memory allocator.

Returns

Pointer to the array of buffers

4.9.2.28 `size_t coap_mem_small_get_len (void)`

Get the length of each buffer in the small memory allocator.

Returns

Length of each buffer

4.9.2.29 `size_t coap_mem_small_get_num (void)`

Get the number of buffers in the small memory allocator.

Returns

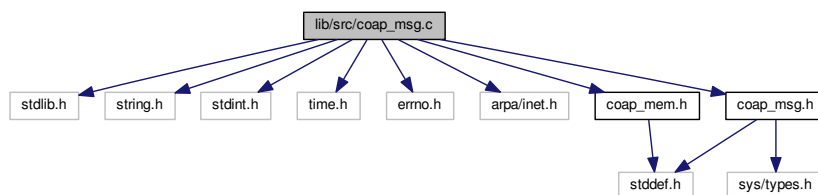
Number of buffers

4.10 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 "coap_mem.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_calc_block_szx` (unsigned size)
Calculate block size exponent from block size.
- 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.
- int `coap_msg_parse_block_op` (unsigned *num, unsigned *more, unsigned *size, coap_msg_t *msg, int type)
Find and parse a Block1 or Block2 option in 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 an option to a message structure.
- int `coap_msg_set_payload` (coap_msg_t *msg, char *buf, size_t len)
Set the payload in a message.
- void `coap_msg_clear_payload` (coap_msg_t *msg)
Clear 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.

4.10.1 Detailed Description

Source file for the FreeCoAP message parser/formatter library.

4.10.2 Macro Definition Documentation

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

Get the first option from an option linked-list

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

Get the last option in an option linked-list

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

Indicate whether or not an option linked-list is empty

4.10.3 Function Documentation

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

Add an option 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

4.10.3.2 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

4.10.3.3 unsigned coap_msg_check_unsafe_ops (coap_msg_t * msg)

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

Parameters

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

Returns

Operation status or bad option number

Return values

0	Success
>0	Bad option number

4.10.3.4 void coap_msg_clear_payload (coap_msg_t * msg)

Clear the payload in a message.

Free the buffer in the message structure containing the current payload if there is one.

Parameters

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

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

Copy a message.

Parameters

in, 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
<0	Error

4.10.3.6 void coap_msg_create (coap_msg_t * *msg*)

Initialise a message structure.

Parameters

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

4.10.3.7 void coap_msg_destroy (coap_msg_t * *msg*)

Deinitialise a message structure.

Parameters

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

4.10.3.8 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

4.10.3.9 void coap_msg_gen_rand_str (char * buf, size_t len)

Generate a random string of bytes.

Parameters

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

4.10.3.10 int coap_msg_op_calc_block_szx (unsigned size)

Calculate block size exponent from block size.

Parameters

in	<i>size</i>	Block size
----	-------------	------------

Returns

Block size exponent or error code

Return values

≥ 0	Block size exponent
< 0	Error

4.10.3.11 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 a buffer to store the option value
in	<i>len</i>	Length of the buffer
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

4.10.3.12 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

4.10.3.13 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

4.10.3.14 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

4.10.3.15 int coap_msg_parse_block_op (unsigned * num, unsigned * more, unsigned * size, coap_msg_t * msg, int type)

Find and parse a Block1 or Block2 option in a message.

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 bytes)
in	<i>msg</i>	Pointer to a message
in	<i>type</i>	Block option type: COAP_MSG_BLOCK1 or COAP_MSG_BLOCK2

Returns

Operation status

Return values

1	Block option not found
0	Success
<0	Error

4.10.3.16 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</i> ↔ <i>_id</i>	Pointer to a field to store the message ID value

Returns

Operation status

Return values

0	Success
<0	Error

4.10.3.17 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
---------	------------	--------------------------------

4.10.3.18 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

4.10.3.19 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

4.10.3.20 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

4.10.3.21 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

4.10.3.22 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

4.10.3.23 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

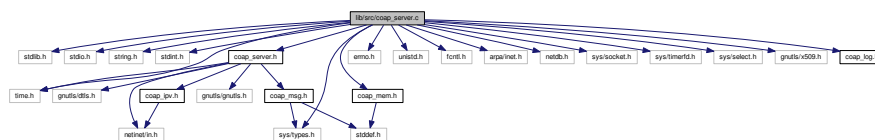
The number of bytes that would be written to the buffer it was large enough

4.11 lib/src/coap_server.c File Reference

Source file for the FreeCoAP server library.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <time.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/timerfd.h>
#include <sys/select.h>
#include <sys/types.h>
#include <gnutls/x509.h>
#include "coap_server.h"
#include "coap_mem.h"
#include "coap_log.h"
```

Include dependency graph for coap_server.c:



Macros

- #define COAP_SERVER_ACK_TIMEOUT_SEC 2
- #define COAP_SERVER_MAX_RETRANSMIT 4
- #define COAP_SERVER_DTLS_MTU COAP_MSG_MAX_BUF_LEN
- #define COAP_SERVER_DTLS_RETRANS_TIMEOUT 1000
- #define COAP_SERVER_DTLS_TOTAL_TIMEOUT 60000
- #define COAP_SERVER_DTLS_HANDSHAKE_ATTEMPTS 60
- #define COAP_SERVER_DTLS_NUM_DH_BITS 1024
- #define COAP_SERVER_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SE<↵
RVER_PRECEDENCE"

Functions

- int `coap_server_trans_handle_blockwise` (`coap_server_trans_t` *trans, `coap_msg_t` *req, `coap_msg_t` *resp, unsigned block1_size, unsigned block2_size, char *body, size_t body_len, `coap_server_trans_handler_t` block_rx)
Handle a library-level blockwise transfer.
- int `coap_server_create` (`coap_server_t` *server, `coap_server_trans_handler_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.
- 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.

4.11.1 Detailed Description

Source file for the FreeCoAP server library.

4.11.2 Macro Definition Documentation

4.11.2.1 #define COAP_SERVER_ACK_TIMEOUT_SEC 2

Minimum delay to wait before retransmitting a confirmable message

4.11.2.2 #define COAP_SERVER_DTLS_HANDSHAKE_ATTEMPTS 60

Maximum number of DTLS handshake attempts

4.11.2.3 #define COAP_SERVER_DTLS_MTU COAP_MSG_MAX_BUF_LEN

Maximum transmission unit excluding the UDP and IPv6 headers

4.11.2.4 #define COAP_SERVER_DTLS_NUM_DH_BITS 1024

DTLS Diffie-Hellman key size

4.11.2.5 #define COAP_SERVER_DTLS_PRIORITIES "PERFORMANCE:-VERS-TLS-ALL:+VERS-DTLS1.0:%SERVER_PRECEDENCE"

DTLS priorities

4.11.2.6 `#define COAP_SERVER_DTLS_RETRANS_TIMEOUT 1000`

Retransmission timeout (msec) for the DTLS handshake

4.11.2.7 `#define COAP_SERVER_DTLS_TOTAL_TIMEOUT 60000`

Total timeout (msec) for the DTLS handshake

4.11.2.8 `#define COAP_SERVER_MAX_RETRANSMIT 4`

Maximum number of times a confirmable message can be retransmitted

4.11.3 Function Documentation

4.11.3.1 `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

4.11.3.2 `int coap_server_create (coap_server_t * server, coap_server_trans_handler_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>	String containing the host address of the server
in	<i>port</i>	String containing the port number of the server
in	<i>key_file_name</i>	String containing the DTLS key file name

Parameters

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

4.11.3.3 void coap_server_destroy (coap_server_t * server)

Deinitialise a server structure.

Parameters

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

4.11.3.4 unsigned coap_server_get_next_msg_id (coap_server_t * server)

Get a new message ID value.

Parameters

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

Returns

message ID value

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

Returns

Operation status

Return values

0	Success
<0	Error

4.11.3.6 int coap_server_trans_handle_blockwise (coap_server_trans_t * *trans*, coap_msg_t * *req*, coap_msg_t * *resp*, unsigned *block1_size*, unsigned *block2_size*, char * *body*, size_t *body_len*, coap_server_trans_handler_t *block_rx*)

Handle a library-level blockwise transfer.

Configure the transaction structure to do a library-level blockwise transfer. This function should be called by the application from the handle callback function.

Parameters

in, out	<i>trans</i>	Pointer to a transaction structure
in	<i>req</i>	Pointer to the request message
out	<i>resp</i>	Pointer to the response message
in	<i>block1_size</i>	Preferred block1 size
in	<i>block2_size</i>	Preferred block2 size
in	<i>body</i>	Buffer containing the body
in	<i>body_len</i>	length of the buffer
in	<i>block_rx</i>	Callback function to be called when the body of a blockwise transfer has been fully received

Returns

Operation status

Return values

0	Success
<0	Error

Index

active
 coap_mem_t, 7
 coap_server_trans, 17

block1_next
 coap_server_trans, 17

block1_size
 coap_server_trans, 17

block2_next
 coap_server_trans, 17

block2_size
 coap_server_trans, 17

block_detail
 coap_server_trans, 17

block_rx
 coap_server_trans, 17

block_uri
 coap_server_trans, 18

body
 coap_server_trans, 18

body_end
 coap_server_trans, 18

body_len
 coap_server_trans, 18

buf
 coap_mem_t, 7

COAP_CLIENT_ACK_TIMEOUT_SEC
 coap_client.c, 63

COAP_CLIENT_DTLS_HANDSHAKE_ATTEMPTS
 coap_client.c, 63

COAP_CLIENT_DTLS_MTU
 coap_client.c, 63

COAP_CLIENT_DTLS_PRIORITIES
 coap_client.c, 63

COAP_CLIENT_DTLS_RETRANS_TIMEOUT
 coap_client.c, 63

COAP_CLIENT_DTLS_TOTAL_TIMEOUT
 coap_client.c, 64

COAP_CLIENT_HOST_BUF_LEN
 coap_client.h, 22

COAP_CLIENT_MAX_RETRANSMIT
 coap_client.c, 64

COAP_CLIENT_PORT_BUF_LEN
 coap_client.h, 22

COAP_CLIENT_RESP_TIMEOUT_SEC
 coap_client.c, 64

COAP_LOG_DEBUG
 coap_log.h, 27

COAP_LOG_DEF_LEVEL
 coap_log.h, 27

COAP_LOG_ERROR
 coap_log.h, 27

COAP_LOG_INFO
 coap_log.h, 27

COAP_LOG_NOTICE
 coap_log.h, 27

COAP_LOG_WARN
 coap_log.h, 27

COAP_MSG_ACCEPT
 coap_msg.h, 46

COAP_MSG_ACK
 coap_msg.h, 47

COAP_MSG_BAD_GATEWAY
 coap_msg.h, 46

COAP_MSG_BAD_OPTION
 coap_msg.h, 45

COAP_MSG_BAD_REQ
 coap_msg.h, 45

COAP_MSG_BLOCK1
 coap_msg.h, 46

COAP_MSG_BLOCK2
 coap_msg.h, 46

COAP_MSG_CHANGED
 coap_msg.h, 47

COAP_MSG_CLIENT_ERR
 coap_msg.h, 45

COAP_MSG_CONTENT_FORMAT
 coap_msg.h, 46

COAP_MSG_CONTENT
 coap_msg.h, 47

COAP_MSG_CONTINUE
 coap_msg.h, 47

COAP_MSG_CON
 coap_msg.h, 47

COAP_MSG_CREATED
 coap_msg.h, 47

COAP_MSG_DELETED
 coap_msg.h, 47

COAP_MSG_DELETE
 coap_msg.h, 46

COAP_MSG_ETAG
 coap_msg.h, 46

COAP_MSG_FORBIDDEN
 coap_msg.h, 45

COAP_MSG_GATEWAY_TIMEOUT
 coap_msg.h, 46

COAP_MSG_GET
 coap_msg.h, 46

COAP_MSG_IF_MATCH
 coap_msg.h, [46](#)

COAP_MSG_IF_NONE_MATCH
 coap_msg.h, [46](#)

COAP_MSG_INCOMPLETE
 coap_msg.h, [45](#)

COAP_MSG_INT_SERVER_ERR
 coap_msg.h, [46](#)

COAP_MSG_LOCATION_PATH
 coap_msg.h, [46](#)

COAP_MSG_LOCATION_QUERY
 coap_msg.h, [46](#)

COAP_MSG_MAX_AGE
 coap_msg.h, [46](#)

COAP_MSG_MAX_BUF_LEN
 coap_msg.h, [43](#)

COAP_MSG_MAX_CODE_CLASS
 coap_msg.h, [43](#)

COAP_MSG_MAX_CODE_DETAIL
 coap_msg.h, [43](#)

COAP_MSG_MAX_MSG_ID
 coap_msg.h, [43](#)

COAP_MSG_MAX_PAYLOAD_LEN
 coap_msg.h, [43](#)

COAP_MSG_MAX_TOKEN_LEN
 coap_msg.h, [43](#)

COAP_MSG_METHOD_NOT_ALLOWED
 coap_msg.h, [45](#)

COAP_MSG_NOT_ACCEPTABLE
 coap_msg.h, [45](#)

COAP_MSG_NOT_FOUND
 coap_msg.h, [45](#)

COAP_MSG_NOT_IMPL
 coap_msg.h, [46](#)

COAP_MSG_NON
 coap_msg.h, [47](#)

COAP_MSG_OP_MAX_BLOCK_SIZE
 coap_msg.h, [44](#)

COAP_MSG_OP_MAX_BLOCK_VAL_LEN
 coap_msg.h, [44](#)

COAP_MSG_OP_URI_PATH_MAX_LEN
 coap_msg.h, [45](#)

COAP_MSG_POST
 coap_msg.h, [46](#)

COAP_MSG_PRECOND_FAILED
 coap_msg.h, [45](#)

COAP_MSG_PROXY_NOT_SUP
 coap_msg.h, [46](#)

COAP_MSG_PROXY_SCHEME
 coap_msg.h, [46](#)

COAP_MSG_PROXY_URI
 coap_msg.h, [46](#)

COAP_MSG_PUT
 coap_msg.h, [46](#)

COAP_MSG_REQ_ENT_TOO_LARGE
 coap_msg.h, [45](#)

COAP_MSG_REQ
 coap_msg.h, [45](#)

COAP_MSG_RST
 coap_msg.h, [47](#)

COAP_MSG_SERV_UNAVAIL
 coap_msg.h, [46](#)

COAP_MSG_SERVER_ERR
 coap_msg.h, [45](#)

COAP_MSG_SIZE1
 coap_msg.h, [46](#)

COAP_MSG_SIZE2
 coap_msg.h, [46](#)

COAP_MSG_SUCCESS
 coap_msg.h, [45](#)

COAP_MSG_UNAUTHORIZED
 coap_msg.h, [45](#)

COAP_MSG_UNSUP_CONT_FMT
 coap_msg.h, [45](#)

COAP_MSG_URI_HOST
 coap_msg.h, [46](#)

COAP_MSG_URI_PATH
 coap_msg.h, [46](#)

COAP_MSG_URI_PORT
 coap_msg.h, [46](#)

COAP_MSG_URI_QUERY
 coap_msg.h, [46](#)

COAP_MSG_VALID
 coap_msg.h, [47](#)

COAP_MSG_VER
 coap_msg.h, [45](#)

COAP_SERVER_ACK_TIMEOUT_SEC
 coap_server.c, [89](#)

COAP_SERVER_ADDR_BUF_LEN
 coap_server.h, [58](#)

COAP_SERVER_DIAG_PAYLOAD_LEN
 coap_server.h, [58](#)

COAP_SERVER_DTLS_HANDSHAKE_ATTEMPTS
 coap_server.c, [89](#)

COAP_SERVER_DTLS_MTU
 coap_server.c, [89](#)

COAP_SERVER_DTLS_NUM_DH_BITS
 coap_server.c, [89](#)

COAP_SERVER_DTLS_PRIORITIES
 coap_server.c, [89](#)

COAP_SERVER_DTLS_RETRANS_TIMEOUT
 coap_server.c, [89](#)

COAP_SERVER_DTLS_TOTAL_TIMEOUT
 coap_server.c, [90](#)

COAP_SERVER_MAX_RETRANSMIT
 coap_server.c, [90](#)

COAP_SERVER_NUM_TRANS
 coap_server.h, [58](#)

COAP_SERVER_PIGGYBACKED
 coap_server.h, [59](#)

COAP_SERVER_SEPARATE
 coap_server.h, [59](#)

COAP_SERVER_TRANS_BLOCKWISE_GET
 coap_server.h, [59](#)

COAP_SERVER_TRANS_BLOCKWISE_POST1
 coap_server.h, [59](#)

- COAP_SERVER_TRANS_BLOCKWISE_POST2
 - coap_server.h, [59](#)
- COAP_SERVER_TRANS_BLOCKWISE_PUT1
 - coap_server.h, [59](#)
- COAP_SERVER_TRANS_BLOCKWISE_PUT2
 - coap_server.h, [59](#)
- COAP_SERVER_TRANS_REGULAR
 - coap_server.h, [59](#)
- client_addr
 - coap_server_trans, [18](#)
- client_sin
 - coap_server_trans, [18](#)
- client_sin_len
 - coap_server_trans, [18](#)
- coap_client.c
 - COAP_CLIENT_ACK_TIMEOUT_SEC, [63](#)
 - COAP_CLIENT_DTLS_HANDSHAKE_ATTEMPTS, [63](#)
 - COAP_CLIENT_DTLS_MTU, [63](#)
 - COAP_CLIENT_DTLS_PRIORITIES, [63](#)
 - COAP_CLIENT_DTLS_RETRANS_TIMEOUT, [63](#)
 - COAP_CLIENT_DTLS_TOTAL_TIMEOUT, [64](#)
 - COAP_CLIENT_MAX_RETRANSMIT, [64](#)
 - COAP_CLIENT_RESP_TIMEOUT_SEC, [64](#)
 - coap_client_create, [64](#)
 - coap_client_destroy, [64](#)
 - coap_client_exchange, [65](#)
 - coap_client_exchange_blockwise, [65](#)
- coap_client.h
 - COAP_CLIENT_HOST_BUF_LEN, [22](#)
 - COAP_CLIENT_PORT_BUF_LEN, [22](#)
 - coap_client_create, [22](#)
 - coap_client_destroy, [23](#)
 - coap_client_exchange, [23](#)
 - coap_client_exchange_blockwise, [24](#)
- coap_client_create
 - coap_client.c, [64](#)
 - coap_client.h, [22](#)
- coap_client_destroy
 - coap_client.c, [64](#)
 - coap_client.h, [23](#)
- coap_client_exchange
 - coap_client.c, [65](#)
 - coap_client.h, [23](#)
- coap_client_exchange_blockwise
 - coap_client.c, [65](#)
 - coap_client.h, [24](#)
- 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, [67](#)
 - coap_log_error, [67](#)
 - coap_log_get_level, [67](#)
 - coap_log_info, [67](#)
 - coap_log_notice, [67](#)
 - coap_log_set_level, [68](#)
 - coap_log_warn, [68](#)
- coap_log.h
 - COAP_LOG_DEBUG, [27](#)
 - COAP_LOG_DEF_LEVEL, [27](#)
 - COAP_LOG_ERROR, [27](#)
 - COAP_LOG_INFO, [27](#)
 - COAP_LOG_NOTICE, [27](#)
 - COAP_LOG_WARN, [27](#)
 - coap_log_debug, [27](#)
 - coap_log_error, [27](#)
 - coap_log_get_level, [27](#)
 - coap_log_info, [28](#)
 - coap_log_level_t, [27](#)
 - coap_log_notice, [28](#)
 - coap_log_set_level, [28](#)
 - coap_log_warn, [28](#)
- coap_log_debug
 - coap_log.c, [67](#)
 - coap_log.h, [27](#)
- coap_log_error
 - coap_log.c, [67](#)
 - coap_log.h, [27](#)
- coap_log_get_level
 - coap_log.c, [67](#)
 - coap_log.h, [27](#)
- coap_log_info
 - coap_log.c, [67](#)
 - coap_log.h, [28](#)
- coap_log_level_t
 - coap_log.h, [27](#)
- coap_log_notice
 - coap_log.c, [67](#)
 - coap_log.h, [28](#)
- coap_log_set_level
 - coap_log.c, [68](#)
 - coap_log.h, [28](#)
- coap_log_warn
 - coap_log.c, [68](#)
 - coap_log.h, [28](#)
- coap_mem.c
 - coap_mem_all_create, [70](#)
 - coap_mem_alloc, [71](#)
 - coap_mem_create, [71](#)
 - coap_mem_destroy, [72](#)
 - coap_mem_free, [72](#)
 - coap_mem_large_alloc, [72](#)
 - coap_mem_large_create, [72](#)
 - coap_mem_large_free, [73](#)
 - coap_mem_large_get_active, [73](#)
 - coap_mem_large_get_active_len, [73](#)

- coap_mem_large_get_buf, 73
- coap_mem_large_get_len, 73
- coap_mem_large_get_num, 74
- coap_mem_medium_alloc, 74
- coap_mem_medium_create, 74
- coap_mem_medium_free, 75
- coap_mem_medium_get_active, 75
- coap_mem_medium_get_active_len, 75
- coap_mem_medium_get_buf, 75
- coap_mem_medium_get_len, 75
- coap_mem_medium_get_num, 75
- coap_mem_small_alloc, 75
- coap_mem_small_create, 76
- coap_mem_small_free, 76
- coap_mem_small_get_active, 76
- coap_mem_small_get_active_len, 76
- coap_mem_small_get_buf, 77
- coap_mem_small_get_len, 77
- coap_mem_small_get_num, 77
- coap_mem.h
 - coap_mem_all_create, 31
 - coap_mem_alloc, 32
 - coap_mem_create, 32
 - coap_mem_destroy, 33
 - coap_mem_free, 33
 - coap_mem_get_active, 31
 - coap_mem_get_active_len, 31
 - coap_mem_get_buf, 31
 - coap_mem_get_len, 31
 - coap_mem_get_num, 31
 - coap_mem_large_alloc, 33
 - coap_mem_large_create, 33
 - coap_mem_large_free, 34
 - coap_mem_large_get_active, 34
 - coap_mem_large_get_active_len, 34
 - coap_mem_large_get_buf, 34
 - coap_mem_large_get_len, 34
 - coap_mem_large_get_num, 35
 - coap_mem_medium_alloc, 35
 - coap_mem_medium_create, 35
 - coap_mem_medium_free, 36
 - coap_mem_medium_get_active, 36
 - coap_mem_medium_get_active_len, 36
 - coap_mem_medium_get_buf, 36
 - coap_mem_medium_get_len, 36
 - coap_mem_medium_get_num, 36
 - coap_mem_small_alloc, 36
 - coap_mem_small_create, 37
 - coap_mem_small_free, 37
 - coap_mem_small_get_active, 37
 - coap_mem_small_get_active_len, 37
 - coap_mem_small_get_buf, 38
 - coap_mem_small_get_len, 38
 - coap_mem_small_get_num, 38
- coap_mem_all_create
 - coap_mem.c, 70
 - coap_mem.h, 31
- coap_mem_alloc
 - coap_mem.c, 71
 - coap_mem.h, 32
- coap_mem_create
 - coap_mem.c, 71
 - coap_mem.h, 32
- coap_mem_destroy
 - coap_mem.c, 72
 - coap_mem.h, 33
- coap_mem_free
 - coap_mem.c, 72
 - coap_mem.h, 33
- coap_mem_get_active
 - coap_mem.h, 31
- coap_mem_get_active_len
 - coap_mem.h, 31
- coap_mem_get_buf
 - coap_mem.h, 31
- coap_mem_get_len
 - coap_mem.h, 31
- coap_mem_get_num
 - coap_mem.h, 31
- coap_mem_large_alloc
 - coap_mem.c, 72
 - coap_mem.h, 33
- coap_mem_large_create
 - coap_mem.c, 72
 - coap_mem.h, 33
- coap_mem_large_free
 - coap_mem.c, 73
 - coap_mem.h, 34
- coap_mem_large_get_active
 - coap_mem.c, 73
 - coap_mem.h, 34
- coap_mem_large_get_active_len
 - coap_mem.c, 73
 - coap_mem.h, 34
- coap_mem_large_get_buf
 - coap_mem.c, 73
 - coap_mem.h, 34
- coap_mem_large_get_len
 - coap_mem.c, 73
 - coap_mem.h, 34
- coap_mem_large_get_num
 - coap_mem.c, 74
 - coap_mem.h, 35
- coap_mem_medium_alloc
 - coap_mem.c, 74
 - coap_mem.h, 35
- coap_mem_medium_create
 - coap_mem.c, 74
 - coap_mem.h, 35
- coap_mem_medium_free
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_medium_get_active
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_medium_get_active_len
 - coap_mem.c, 75
 - coap_mem.h, 36

- coap_mem.c, 75
- coap_mem.h, 36
- coap_mem_medium_get_buf
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_medium_get_len
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_medium_get_num
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_small_alloc
 - coap_mem.c, 75
 - coap_mem.h, 36
- coap_mem_small_create
 - coap_mem.c, 76
 - coap_mem.h, 37
- coap_mem_small_free
 - coap_mem.c, 76
 - coap_mem.h, 37
- coap_mem_small_get_active
 - coap_mem.c, 76
 - coap_mem.h, 37
- coap_mem_small_get_active_len
 - coap_mem.c, 76
 - coap_mem.h, 37
- coap_mem_small_get_buf
 - coap_mem.c, 77
 - coap_mem.h, 38
- coap_mem_small_get_len
 - coap_mem.c, 77
 - coap_mem.h, 38
- coap_mem_small_get_num
 - coap_mem.c, 77
 - coap_mem.h, 38
- coap_mem_t, 7
 - active, 7
 - buf, 7
 - len, 7
 - num, 7
- coap_msg.c
 - coap_msg_add_op, 79
 - coap_msg_check_critical_ops, 80
 - coap_msg_check_unsafe_ops, 80
 - coap_msg_clear_payload, 80
 - coap_msg_copy, 80
 - coap_msg_create, 81
 - coap_msg_destroy, 81
 - coap_msg_format, 81
 - coap_msg_gen_rand_str, 82
 - coap_msg_op_calc_block_szx, 82
 - coap_msg_op_format_block_val, 82
 - coap_msg_op_list_get_first, 79
 - coap_msg_op_list_get_last, 79
 - coap_msg_op_list_is_empty, 79
 - coap_msg_op_num_is_recognized, 83
 - coap_msg_op_parse_block_val, 83
 - coap_msg_parse, 84
 - coap_msg_parse_block_op, 84
 - coap_msg_parse_type_msg_id, 84
 - coap_msg_reset, 85
 - coap_msg_set_code, 85
 - coap_msg_set_msg_id, 86
 - coap_msg_set_payload, 86
 - coap_msg_set_token, 86
 - coap_msg_set_type, 87
 - coap_msg_uri_path_to_str, 87
- coap_msg.h
 - COAP_MSG_ACCEPT, 46
 - COAP_MSG_ACK, 47
 - COAP_MSG_BAD_GATEWAY, 46
 - COAP_MSG_BAD_OPTION, 45
 - COAP_MSG_BAD_REQ, 45
 - COAP_MSG_BLOCK1, 46
 - COAP_MSG_BLOCK2, 46
 - COAP_MSG_CHANGED, 47
 - COAP_MSG_CLIENT_ERR, 45
 - COAP_MSG_CONTENT_FORMAT, 46
 - COAP_MSG_CONTENT, 47
 - COAP_MSG_CONTINUE, 47
 - COAP_MSG_CON, 47
 - COAP_MSG_CREATED, 47
 - COAP_MSG_DELETED, 47
 - COAP_MSG_DELETE, 46
 - COAP_MSG_ETAG, 46
 - COAP_MSG_FORBIDDEN, 45
 - COAP_MSG_GATEWAY_TIMEOUT, 46
 - COAP_MSG_GET, 46
 - COAP_MSG_IF_MATCH, 46
 - COAP_MSG_IF_NONE_MATCH, 46
 - COAP_MSG_INCOMPLETE, 45
 - COAP_MSG_INT_SERVER_ERR, 46
 - COAP_MSG_LOCATION_PATH, 46
 - COAP_MSG_LOCATION_QUERY, 46
 - COAP_MSG_MAX_AGE, 46
 - COAP_MSG_MAX_BUF_LEN, 43
 - COAP_MSG_MAX_CODE_CLASS, 43
 - COAP_MSG_MAX_CODE_DETAIL, 43
 - COAP_MSG_MAX_MSG_ID, 43
 - COAP_MSG_MAX_PAYLOAD_LEN, 43
 - COAP_MSG_MAX_TOKEN_LEN, 43
 - COAP_MSG_METHOD_NOT_ALLOWED, 45
 - COAP_MSG_NOT_ACCEPTABLE, 45
 - COAP_MSG_NOT_FOUND, 45
 - COAP_MSG_NOT_IMPL, 46
 - COAP_MSG_NON, 47
 - COAP_MSG_OP_MAX_BLOCK_SIZE, 44
 - COAP_MSG_OP_MAX_BLOCK_VAL_LEN, 44
 - COAP_MSG_OP_URI_PATH_MAX_LEN, 45
 - COAP_MSG_POST, 46
 - COAP_MSG_PRECOND_FAILED, 45
 - COAP_MSG_PROXY_NOT_SUP, 46
 - COAP_MSG_PROXY_SCHEME, 46
 - COAP_MSG_PROXY_URI, 46
 - COAP_MSG_PUT, 46
 - COAP_MSG_REQ_ENT_TOO_LARGE, 45

COAP_MSG_REQ, 45
 COAP_MSG_RST, 47
 COAP_MSG_SERV_UNAVAIL, 46
 COAP_MSG_SERVER_ERR, 45
 COAP_MSG_SIZE1, 46
 COAP_MSG_SIZE2, 46
 COAP_MSG_SUCCESS, 45
 COAP_MSG_UNAUTHORIZED, 45
 COAP_MSG_UNSUP_CONT_FMT, 45
 COAP_MSG_URI_HOST, 46
 COAP_MSG_URI_PATH, 46
 COAP_MSG_URI_PORT, 46
 COAP_MSG_URI_QUERY, 46
 COAP_MSG_VALID, 47
 COAP_MSG_VER, 45
 coap_msg_add_op, 47
 coap_msg_block_start_to_num, 42
 coap_msg_block_szx_to_size, 42
 coap_msg_check_critical_ops, 47
 coap_msg_check_unsafe_ops, 48
 coap_msg_class_t, 45
 coap_msg_clear_payload, 48
 coap_msg_client_err_t, 45
 coap_msg_copy, 48
 coap_msg_create, 49
 coap_msg_destroy, 49
 coap_msg_format, 49
 coap_msg_gen_rand_str, 50
 coap_msg_get_code_class, 42
 coap_msg_get_code_detail, 42
 coap_msg_get_first_op, 42
 coap_msg_get_msg_id, 42
 coap_msg_get_payload, 42
 coap_msg_get_payload_len, 42
 coap_msg_get_token, 42
 coap_msg_get_token_len, 42
 coap_msg_get_type, 43
 coap_msg_get_ver, 43
 coap_msg_is_empty, 43
 coap_msg_method_t, 45
 coap_msg_op_calc_block_szx, 50
 coap_msg_op_format_block_val, 50
 coap_msg_op_get_len, 43
 coap_msg_op_get_next, 44
 coap_msg_op_get_num, 44
 coap_msg_op_get_val, 44
 coap_msg_op_num_is_critical, 44
 coap_msg_op_num_is_recognized, 51
 coap_msg_op_num_is_unsafe, 44
 coap_msg_op_num_no_cache_key, 44
 coap_msg_op_num_t, 46
 coap_msg_op_parse_block_val, 51
 coap_msg_op_set_len, 44
 coap_msg_op_set_next, 44
 coap_msg_op_set_num, 45
 coap_msg_op_set_val, 45
 coap_msg_parse, 52
 coap_msg_parse_block_op, 52
 coap_msg_parse_type_msg_id, 52
 coap_msg_reset, 53
 coap_msg_server_err_t, 46
 coap_msg_set_code, 53
 coap_msg_set_msg_id, 54
 coap_msg_set_payload, 54
 coap_msg_set_token, 54
 coap_msg_set_type, 55
 coap_msg_success_t, 46
 coap_msg_type_t, 47
 coap_msg_uri_path_to_str, 55
 coap_msg_add_op
 coap_msg.c, 79
 coap_msg.h, 47
 coap_msg_block_start_to_num
 coap_msg.h, 42
 coap_msg_block_szx_to_size
 coap_msg.h, 42
 coap_msg_check_critical_ops
 coap_msg.c, 80
 coap_msg.h, 47
 coap_msg_check_unsafe_ops
 coap_msg.c, 80
 coap_msg.h, 48
 coap_msg_class_t
 coap_msg.h, 45
 coap_msg_clear_payload
 coap_msg.c, 80
 coap_msg.h, 48
 coap_msg_client_err_t
 coap_msg.h, 45
 coap_msg_copy
 coap_msg.c, 80
 coap_msg.h, 48
 coap_msg_create
 coap_msg.c, 81
 coap_msg.h, 49
 coap_msg_destroy
 coap_msg.c, 81
 coap_msg.h, 49
 coap_msg_format
 coap_msg.c, 81
 coap_msg.h, 49
 coap_msg_gen_rand_str
 coap_msg.c, 82
 coap_msg.h, 50
 coap_msg_get_is_code_class
 coap_msg.h, 42
 coap_msg_get_code_detail
 coap_msg.h, 42
 coap_msg_get_first_op
 coap_msg.h, 42
 coap_msg_get_msg_id
 coap_msg.h, 42
 coap_msg_get_payload
 coap_msg.h, 42
 coap_msg_get_payload_len
 coap_msg.h, 42

coap_msg_get_token
 coap_msg.h, [42](#)
coap_msg_get_token_len
 coap_msg.h, [42](#)
coap_msg_get_type
 coap_msg.h, [43](#)
coap_msg_get_ver
 coap_msg.h, [43](#)
coap_msg_is_empty
 coap_msg.h, [43](#)
coap_msg_method_t
 coap_msg.h, [45](#)
coap_msg_op, [8](#)
 len, [8](#)
 next, [8](#)
 num, [8](#)
 val, [8](#)
coap_msg_op_calc_block_szx
 coap_msg.c, [82](#)
 coap_msg.h, [50](#)
coap_msg_op_format_block_val
 coap_msg.c, [82](#)
 coap_msg.h, [50](#)
coap_msg_op_get_len
 coap_msg.h, [43](#)
coap_msg_op_get_next
 coap_msg.h, [44](#)
coap_msg_op_get_num
 coap_msg.h, [44](#)
coap_msg_op_get_val
 coap_msg.h, [44](#)
coap_msg_op_list_get_first
 coap_msg.c, [79](#)
coap_msg_op_list_get_last
 coap_msg.c, [79](#)
coap_msg_op_list_is_empty
 coap_msg.c, [79](#)
coap_msg_op_list_t, [9](#)
 first, [9](#)
 last, [9](#)
coap_msg_op_num_is_critical
 coap_msg.h, [44](#)
coap_msg_op_num_is_recognized
 coap_msg.c, [83](#)
 coap_msg.h, [51](#)
coap_msg_op_num_is_unsafe
 coap_msg.h, [44](#)
coap_msg_op_num_no_cache_key
 coap_msg.h, [44](#)
coap_msg_op_num_t
 coap_msg.h, [46](#)
coap_msg_op_parse_block_val
 coap_msg.c, [83](#)
 coap_msg.h, [51](#)
coap_msg_op_set_len
 coap_msg.h, [44](#)
coap_msg_op_set_next
 coap_msg.h, [44](#)
coap_msg_op_set_num
 coap_msg.h, [45](#)
coap_msg_op_set_val
 coap_msg.h, [45](#)
coap_msg_parse
 coap_msg.c, [84](#)
 coap_msg.h, [52](#)
coap_msg_parse_block_op
 coap_msg.c, [84](#)
 coap_msg.h, [52](#)
coap_msg_parse_type_msg_id
 coap_msg.c, [84](#)
 coap_msg.h, [52](#)
coap_msg_reset
 coap_msg.c, [85](#)
 coap_msg.h, [53](#)
coap_msg_server_err_t
 coap_msg.h, [46](#)
coap_msg_set_code
 coap_msg.c, [85](#)
 coap_msg.h, [53](#)
coap_msg_set_msg_id
 coap_msg.c, [86](#)
 coap_msg.h, [54](#)
coap_msg_set_payload
 coap_msg.c, [86](#)
 coap_msg.h, [54](#)
coap_msg_set_token
 coap_msg.c, [86](#)
 coap_msg.h, [54](#)
coap_msg_set_type
 coap_msg.c, [87](#)
 coap_msg.h, [55](#)
coap_msg_success_t
 coap_msg.h, [46](#)
coap_msg_t, [10](#)
 code_class, [11](#)
 code_detail, [11](#)
 msg_id, [11](#)
 op_list, [11](#)
 payload, [11](#)
 payload_len, [11](#)
 token, [11](#)
 token_len, [11](#)
 type, [11](#)
 ver, [11](#)
coap_msg_type_t
 coap_msg.h, [47](#)
coap_msg_uri_path_to_str
 coap_msg.c, [87](#)
 coap_msg.h, [55](#)
coap_server, [12](#)
 cred, [13](#)
 dh_params, [13](#)
 handle, [13](#)
 msg_id, [13](#)
 priority, [13](#)
 sd, [13](#)

- sep_list, [13](#)
- trans, [13](#)
- coap_server.c
 - COAP_SERVER_ACK_TIMEOUT_SEC, [89](#)
 - COAP_SERVER_DTLS_HANDSHAKE_ATTEMPTS, [89](#)
 - COAP_SERVER_DTLS_MTU, [89](#)
 - COAP_SERVER_DTLS_NUM_DH_BITS, [89](#)
 - COAP_SERVER_DTLS_PRIORITIES, [89](#)
 - COAP_SERVER_DTLS_RETRANS_TIMEOUT, [89](#)
 - COAP_SERVER_DTLS_TOTAL_TIMEOUT, [90](#)
 - COAP_SERVER_MAX_RETRANSMIT, [90](#)
 - coap_server_add_sep_resp_uri_path, [90](#)
 - coap_server_create, [90](#)
 - coap_server_destroy, [91](#)
 - coap_server_get_next_msg_id, [91](#)
 - coap_server_run, [91](#)
 - coap_server_trans_handle_blockwise, [92](#)
- coap_server.h
 - COAP_SERVER_ADDR_BUF_LEN, [58](#)
 - COAP_SERVER_DIAG_PAYLOAD_LEN, [58](#)
 - COAP_SERVER_NUM_TRANS, [58](#)
 - COAP_SERVER_PIGGYBACKED, [59](#)
 - COAP_SERVER_SEPARATE, [59](#)
 - COAP_SERVER_TRANS_BLOCKWISE_GET, [59](#)
 - COAP_SERVER_TRANS_BLOCKWISE_POST1, [59](#)
 - COAP_SERVER_TRANS_BLOCKWISE_POST2, [59](#)
 - COAP_SERVER_TRANS_BLOCKWISE_PUT1, [59](#)
 - COAP_SERVER_TRANS_BLOCKWISE_PUT2, [59](#)
 - COAP_SERVER_TRANS_REGULAR, [59](#)
 - coap_server_add_sep_resp_uri_path, [59](#)
 - coap_server_create, [60](#)
 - coap_server_destroy, [60](#)
 - coap_server_get_next_msg_id, [61](#)
 - coap_server_resp_t, [59](#)
 - coap_server_run, [61](#)
 - coap_server_trans_get_body, [58](#)
 - coap_server_trans_get_body_end, [58](#)
 - coap_server_trans_get_body_len, [58](#)
 - coap_server_trans_get_req, [58](#)
 - coap_server_trans_get_resp, [58](#)
 - coap_server_trans_get_type, [58](#)
 - coap_server_trans_handle_blockwise, [61](#)
 - coap_server_trans_handler_t, [58](#)
 - coap_server_trans_set_body_end, [58](#)
 - coap_server_trans_type_t, [59](#)
- coap_server_add_sep_resp_uri_path
 - coap_server.c, [90](#)
 - coap_server.h, [59](#)
- coap_server_create
 - coap_server.c, [90](#)
 - coap_server.h, [60](#)
- coap_server_destroy
 - coap_server.c, [91](#)
 - coap_server.h, [60](#)
- coap_server_get_next_msg_id
 - coap_server.c, [91](#)
 - coap_server.h, [61](#)
- coap_server_path, [14](#)
 - next, [14](#)
 - str, [14](#)
- coap_server_path_list_t, [15](#)
 - first, [15](#)
 - last, [15](#)
- coap_server_resp_t
 - coap_server.h, [59](#)
- coap_server_run
 - coap_server.c, [91](#)
 - coap_server.h, [61](#)
- coap_server_trans, [16](#)
 - active, [17](#)
 - block1_next, [17](#)
 - block1_size, [17](#)
 - block2_next, [17](#)
 - block2_size, [17](#)
 - block_detail, [17](#)
 - block_rx, [17](#)
 - block_uri, [18](#)
 - body, [18](#)
 - body_end, [18](#)
 - body_len, [18](#)
 - client_addr, [18](#)
 - client_sin, [18](#)
 - client_sin_len, [18](#)
 - last_use, [18](#)
 - num_retrans, [18](#)
 - req, [18](#)
 - resp, [19](#)
 - server, [19](#)
 - session, [19](#)
 - timeout, [19](#)
 - timer_fd, [19](#)
 - type, [19](#)
- coap_server_trans_get_body
 - coap_server.h, [58](#)
- coap_server_trans_get_body_end
 - coap_server.h, [58](#)
- coap_server_trans_get_body_len
 - coap_server.h, [58](#)
- coap_server_trans_get_req
 - coap_server.h, [58](#)
- coap_server_trans_get_resp
 - coap_server.h, [58](#)
- coap_server_trans_get_type
 - coap_server.h, [58](#)
- coap_server_trans_handle_blockwise
 - coap_server.c, [92](#)
 - coap_server.h, [61](#)
- coap_server_trans_handler_t
 - coap_server.h, [58](#)
- coap_server_trans_set_body_end

- coap_server.h, [58](#)
- coap_server_trans_type_t
 - coap_server.h, [59](#)
- code_class
 - coap_msg_t, [11](#)
- code_detail
 - coap_msg_t, [11](#)
- cred
 - coap_client_t, [5](#)
 - coap_server, [13](#)
- dh_params
 - coap_server, [13](#)
- first
 - coap_msg_op_list_t, [9](#)
 - coap_server_path_list_t, [15](#)
- handle
 - coap_server, [13](#)
- last
 - coap_msg_op_list_t, [9](#)
 - coap_server_path_list_t, [15](#)
- last_use
 - coap_server_trans, [18](#)
- len
 - coap_mem_t, [7](#)
 - coap_msg_op, [8](#)
- lib/include/coap_client.h, [21](#)
- lib/include/coap_ipv.h, [24](#)
- lib/include/coap_log.h, [26](#)
- lib/include/coap_mem.h, [29](#)
- lib/include/coap_msg.h, [38](#)
- lib/include/coap_server.h, [56](#)
- lib/src/coap_client.c, [62](#)
- lib/src/coap_log.c, [66](#)
- lib/src/coap_mem.c, [68](#)
- lib/src/coap_msg.c, [77](#)
- lib/src/coap_server.c, [88](#)
- msg_id
 - coap_msg_t, [11](#)
 - coap_server, [13](#)
- next
 - coap_msg_op, [8](#)
 - coap_server_path, [14](#)
- num
 - coap_mem_t, [7](#)
 - coap_msg_op, [8](#)
- num_retrans
 - coap_client_t, [5](#)
 - coap_server_trans, [18](#)
- op_list
 - coap_msg_t, [11](#)
- payload
 - coap_msg_t, [11](#)
- payload_len
 - coap_msg_t, [11](#)
- priority
 - coap_client_t, [6](#)
 - coap_server, [13](#)
- req
 - coap_server_trans, [18](#)
- resp
 - coap_server_trans, [19](#)
- sd
 - coap_client_t, [6](#)
 - coap_server, [13](#)
- sep_list
 - coap_server, [13](#)
- server
 - coap_server_trans, [19](#)
- 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, [19](#)
- str
 - coap_server_path, [14](#)
- timeout
 - coap_client_t, [6](#)
 - coap_server_trans, [19](#)
- timer_fd
 - coap_client_t, [6](#)
 - coap_server_trans, [19](#)
- token
 - coap_msg_t, [11](#)
- token_len
 - coap_msg_t, [11](#)
- trans
 - coap_server, [13](#)
- type
 - coap_msg_t, [11](#)
 - coap_server_trans, [19](#)
- val
 - coap_msg_op, [8](#)
- ver
 - coap_msg_t, [11](#)