NAME

ares_dns_class_t, ares_dns_flags_t, ares_dns_opcode_t, ares_dns_parse, ares_dns_rcode_t, ares_dns_record_create, ares_dns_record_destroy, ares_dns_record_get_flags, ares_dns_record_get_id, ares_dns_record_get_opcode, ares_dns_record_get_rcode, ares_dns_record_query_add, ares_dns_record_query_cnt, ares_dns_record_query_get, ares_dns_rec_type_t, ares_dns_write − DNS Record parsing, writing, creating and destroying functions.

SYNOPSIS

```
#include <ares.h>

void ares_dns_record_destroy(ares_dns_record_t *dnsrec);

ares_status_t ares_dns_parse(const unsigned char *buf,
                             size_t buf_len, unsigned int flags,
                             ares_dns_record_t **dnsrec);

ares_status_t ares_dns_write(const ares_dns_record_t *dnsrec,
                             unsigned char **buf, size_t *buf_len);

ares_status_t ares_dns_record_create(ares_dns_record_t **dnsrec,
                                     unsigned short      id,
                                     unsigned short      flags,
                                     ares_dns_opcode_t   opcode,
                                     ares_dns_rcode_t    rcode);

ares_dns_record_t *ares_dns_record_duplicate(const ares_dns_record_t *dnsrec);

unsigned short ares_dns_record_get_id(const ares_dns_record_t *dnsrec);

ares_bool_t ares_dns_record_set_id(ares_dns_record_t *dnsrec,
                                   unsigned short id);

unsigned short ares_dns_record_get_flags(const ares_dns_record_t *dnsrec);

ares_dns_opcode_t ares_dns_record_get_opcode(const ares_dns_record_t *dnsrec);

ares_dns_rcode_t ares_dns_record_get_rcode(const ares_dns_record_t *dnsrec);

ares_status_t ares_dns_record_query_add(ares_dns_record_t  *dnsrec,
                                        const char         *name,
                                        ares_dns_rec_type_t qtype,
                                        ares_dns_class_t    qclass);

ares_status_t ares_dns_record_query_set_name(ares_dns_record_t  *dnsrec,
                                             size_t              idx,
                                             const char         *name);

ares_status_t ares_dns_record_query_set_type(ares_dns_record_t  *dnsrec,
                                             size_t              idx,
                                             ares_dns_rec_type_t qtype);

size_t ares_dns_record_query_cnt(const ares_dns_record_t *dnsrec);

ares_status_t ares_dns_record_query_get(const ares_dns_record_t *dnsrec,
                                        size_t idx, const char **name,
```

```
                    ares_dns_rec_type_t *qtype,
                    ares_dns_class_t *qclass);
```

ENUMERATIONS

ares_dns_rec_type_t - DNS Record types handled by c-ares. Some record types may only be valid on requests, and some may only be valid on responses:

ARES_REC_TYPE_A - Host address
ARES_REC_TYPE_NS - Authoritative server
ARES_REC_TYPE_CNAME - Canonical name
ARES_REC_TYPE_SOA - Start of authority zone
ARES_REC_TYPE_PTR - Domain name pointer
ARES_REC_TYPE_HINFO - Host information
ARES_REC_TYPE_MX - Mail routing information
ARES_REC_TYPE_TXT - Text strings
ARES_REC_TYPE_SIG - RFC 2535. RFC 2931. SIG Record
ARES_REC_TYPE_AAAA - RFC 3596. Ip6 Address
ARES_REC_TYPE_SRV - RFC 2782. Server Selection
ARES_REC_TYPE_NAPTR - RFC 3403. Naming Authority Pointer
ARES_REC_TYPE_OPT - RFC 6891. EDNS0 option (meta-RR). Pseudo Record.
ARES_REC_TYPE_TLSA - RFC 6698. DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA
ARES_REC_TYPE_SVCB - RFC 9460. General Purpose Service Binding
ARES_REC_TYPE_HTTPS - - RFC 9460. Service Binding type for use with HTTPS
ARES_REC_TYPE_ANY - Wildcard match. Not response RR
ARES_REC_TYPE_URI - RFC 7553. Uniform Resource Identifier
ARES_REC_TYPE_CAA - RFC 6844. Certification Authority Authorization
ARES_REC_TYPE_RAW_RR - Used as an indicator that the RR record is not parsed, but provided in wire format

ares_dns_class_t - DNS Classes for requests and responses:

ARES_CLASS_IN - Internet
ARES_CLASS_CHAOS - CHAOS
ARES_CLASS_HESOID - Hesoid [Dyer 87]
ARES_CLASS_NONE - RFC 2136
ARES_CLASS_ANY - Any class (requests only)

ares_dns_opcode_t - DNS Header Opcodes:

ARES_OPCODE_QUERY - Standard query
ARES_OPCODE_IQUERY - Inverse query. Obsolete
ARES_OPCODE_STATUS - Name server status query
ARES_OPCODE_NOTIFY - Zone change notification (RFC 1996)
ARES_OPCODE_UPDATE - Zone update message (RFC 2136)

ares_dns_flags_t - DNS Header Flags:

ARES_FLAG_QR - QR. If set, is a response
ARES_FLAG_AA - Authoritative Answer. If set, is authoritative
ARES_FLAG_TC - Truncation. If set, is truncated response
ARES_FLAG_RD - Recursion Desired. If set, recursion is desired
ARES_FLAG_RA - Recursion Available. If set, server supports recursion
ARES_FLAG_AD - RFC 2065. Authentic Data bit indicates in a response that the data included has been verified by the server providing it
ARES_FLAG_CD - RFC 2065. Checking Disabled bit indicates in a query that non-verified data is acceptable to the resolver sending the query

ares_dns_rcode_t - DNS Response codes from server:

ARES_RCODE_NOERROR - Success

ARES_RCODE_FORMERR - Format error. The name server was unable to interpret the query

ARES_RCODE_SERVFAIL - Server Failure. The name server was unable to process this query due to a problem with the nameserver

ARES_RCODE_NXDOMAIN - Name Error. Meaningful only for responses from an authoritative name server, this code signifies that the domain name referenced in the query does not exist.

ARES_RCODE_NOTIMP - Not implemented. The name server does not support the requested kind of query

ARES_RCODE_REFUSED - Refused. The name server refuses to perform the specified operation for policy reasons.

ARES_RCODE_YXDOMAIN - RFC 2136. Some name that ought not to exist, does exist

ARES_RCODE_YXRRSET - RFC 2136. Some RRset that ought to not exist, does exist

ARES_RCODE_NXRRSET - RFC 2136. Some RRset that ought to exist, does not exist

ARES_RCODE_NOTAUTH - RFC 2136. The server is not authoritative for the zone named in the Zone section.

ARES_RCODE_NOTZONE - RFC 2136. A name used in the Prerequisite or Update Section is not within the zone denoted by the Zone Section.

ARES_RCODE_DSOTYPEI - RFC 8409. DSO-TYPE Not implemented

ARES_RCODE_BADSIG - RFC 8945. TSIG Signature Failure

ARES_RCODE_BADKEY - RFC 8945. Key not recognized

ARES_RCODE_BADTIME - RFC 8945. Signature out of time window

ARES_RCODE_BADMODE - RFC 2930. Bad TKEY Mode

ARES_RCODE_BADNAME - RFC 2930. Duplicate Key Name

ARES_RCODE_BADALG - RFC 2930. Algorithm not supported

ARES_RCODE_BADTRUNC - RFC 8945. Bad Truncation

ARES_RCODE_BADCOOKIE - RFC 7973. Bad/missing Server Cookie

ares_dns_parse_flags_t - Flags for altering *ares_dns_parse(3)* behaviour:

ARES_DNS_PARSE_AN_BASE_RAW - Parse Answer Section from RFC 1035 that allow name compression as RAW RR type

ARES_DNS_PARSE_NS_BASE_RAW - Parse Authority Section from RFC 1035 that allow name compression as RAW RR type

ARES_DNS_PARSE_AR_BASE_RAW - Parse Additional Section from RFC 1035 that allow name compression as RAW RR type

ARES_DNS_PARSE_AN_EXT_RAW - Parse Answer Section from later RFCs (no name compression) as RAW RR type

ARES_DNS_PARSE_NS_EXT_RAW - Parse Authority Section from later RFCs (no name compression) as RAW RR type

ARES_DNS_PARSE_AR_EXT_RAW - Parse Additional Section from later RFCs (no name compression) as RAW RR type

DESCRIPTION

The *ares_dns_record_destroy(3)* function destroys the memory associated with the dns record created by either *ares_dns_record_create(3)* or *ares_dns_parse(3)* passed in via *dnsrec*.

The *ares_dns_parse(3)* function parses the buffer provided in *buf* with length provided in *buf_len.* The *flags* parameter can be one or more *ares_dns_parse_flags_t*, or zero if no flags are needed. The resulting dns record data structure is stored into the variable pointed to by *dnsrec* and must be destroyed using *ares_dns_record_destroy(3)*.

The *ares_dns_write(3)* function takes a populated DNS record structure in *dnsrec* and writes a wire-format DNS message into the variable pointed to by *buf* and writes the length of the buffer into the variable pointed to by *buf_len.* The buffer must be destroyed using *ares_free_string(3).*

The *ares_dns_record_create(3)* function creates an empty DNS record structure in the variable pointed to by *dnsrec*. The *id* parameter is the DNS message id, however if passing to *ares_send(3)* this identifier will be overwritten, so should typically be 0. The *flags* parameter is one or more *ares_dns_flags_t*. The opcode is passed in the *opcode* parameter and should typically be *ARES_OPCODE_QUERY*. The response code is meant mostly for responses and is passed in the *rcode* parameter and is typically *ARES_RCODE_NOERROR*.

The *ares_dns_record_duplicate(3)* function duplicates an existing DNS record structure. This may be useful if needing to save a result as retrieved from *ares_send_dnsrec(3)* or *ares_search_dnsrec(3)*. The structure to be duplicated is passed in the *dnsrec* parameter, and the duplicated copy is returned, or NULL on error such as out of memory.

The *ares_dns_record_get_id(3)* function is used to retrieve the DNS message id from the DNS record provided in the *dnsrec* parameter.

The *ares_dns_record_set_id(3)* function is used to set the DNS message id in the *id* parameter from the DNS record provided in the *dnsrec* parameter. This id will be overwritten when passing the record to c-ares, so mostly exists for external purposes.

The *ares_dns_record_get_flags(3)* function is used to retrieve the DNS message flags from the DNS record provided in the *dnsrec* parameter.

The *ares_dns_record_get_opcode(3)* function is used to retrieve the DNS message flags from the DNS record provided in the *dnsrec* parameter.

The *ares_dns_record_get_rcode(3)* function is used to retrieve the DNS message response code from the DNS record provided in the *dnsrec* parameter.

The *ares_dns_record_query_add(3)* function is used to add a question to the DNS record provided in the *dnsrec* parameter. The domain name specified for the question is provided in the *name* parameter, along with the question type in the *qtype* parameter and the question class (typically *ARES_CLASS_IN*) in the *qclass* parameter.

The *ares_dns_record_query_set_name(3)* function is used to modify the question name in the DNS record provided in the *dnsrec* parameter. The index of the query, which must be less than *ares_dns_record_query_cnt(3)*, is provided in the *idx* parameter. The new domain name is provided in the *name* parameter. Care should be taken as this will cause invalidation of any *name* pointer retrieved from *ares_dns_Record_query_get(3)*. This function is useful if sending multiple similar queries without re-creating the entire DNS query.

The *ares_dns_record_query_set_type(3)* function is used to modify the question type in the DNS record provided in the *dnsrec* parameter. The index of the query, which must be less than *ares_dns_record_query_cnt(3)*, is provided in the *idx* parameter. The new query type is provided in the *qtype* parameter.

The *ares_dns_record_query_cnt(3)* function is used to retrieve the number of DNS questions in the DNS record provided in the *dnsrec* parameter.

The *ares_dns_record_query_get(3)* function is used to retrieve the details of a single DNS question in the provided *dnsrec* parameter. The index provided in the *idx* parameter must be less than the value returned from *ares_dns_record_query_cnt(3)*. The DNS question name will be returned in the variable pointed to by the *name* parameter, this may be provided as NULL if the name is not needed. This pointer will be invalided by any call to *ares_dns_record_query_set_name(3)*. The DNS question type will be returned in the

variable pointed to by the *qtype* parameter, this may be provided as NULL if the type is not needed. The DNS question class will be returned in the variable pointed to by the *qclass* parameter, this may be provided as NULL if the class is not needed.

RETURN VALUES

ares_dns_parse(3), *ares_dns_write(3)*, *ares_dns_record_create(3)*, *ares_dns_record_query_add(3)*, and *ares_dns_record_query_get(3)* all return an *ares_status_t* error code. **ARES_SUCCESS** is returned on success, **ARES_ENOMEM** is returned on out of memory, **ARES_EFORMERR** is returned on misuse.

ares_dns_record_get_id(3), *ares_dns_record_set_id(3)*, *ares_dns_record_get_flags(3)*, *ares_dns_record_get_opcode(3)*, *ares_dns_record_get_rcode(3)*, and *ares_dns_record_query_cnt(3)* all returned their prescribed datatype values and in general can't fail except for misuse cases, in which a 0 may be returned, however 0 can also be a valid return value for most of these functions.

AVAILABILITY

These functions were first introduced in c-ares version 1.22.0.

SEE ALSO

ares_dns_mapping(3), **ares_dns_rr**(3), **ares_free_string**(3)