pycares Documentation

Release 2.3.0

Saúl Ibarra Corretgé

Aug 13, 2018

Contents

Python interface for c-ares.

pycares is a Python module which provides an interface to c-ares. c-ares (http://c-ares.haxx.se/) c-ares is a C library that performs DNS requests and name resolves asynchronously.

CHAPTER 1

Contents

1.1 pycares — Python interface to c-ares.

See also:

c-ares source code.

1.1.1 Objects

Channel - Ares Channel

class pycares.**Channel**([*flags*, *timeout*, *tries*, *ndots*, *tcp_port*, *udp_port*, *servers*, *domains*, *lookups*, *sock_state_cb*, *socket_send_buffer_size*, *socket_receive_buffer_size*, *rotate*])

Parameters

- **flags** (*int*) – Flags controlling the behavior of the resolver. See `constants` for available values.
- **timeout** (*float*) – The number of seconds each name server is given to respond to a query on the first try. The default is five seconds.
- **tries** (*int*) – The number of tries the resolver will try contacting each name server before giving up. The default is four tries.
- **ndots** (*int*) – The number of dots which must be present in a domain name for it to be queried for "as is" prior to querying for it with the default domain extensions appended. The default value is 1 unless set otherwise by resolv.conf or the RES_OPTIONS environment variable.
- **tcp_port** (*int*) – The (TCP) port to use for queries. The default is 53.
- **udp_port** (*int*) – The (UDP) port to use for queries. The default is 53.
- **servers** (*list*) – List of nameservers to be used to do the lookups.

- **domains** (*list*) – The domains to search, instead of the domains specified in resolv.conf or the domain derived from the kernel hostname variable.
- **lookup** (*str*) – The lookups to perform for host queries. lookups should be set to a string of the characters "b" or "f", where "b" indicates a DNS lookup and "f" indicates a lookup in the hosts file.
- **sock_state_cb** (*callable*) – A callback function to be invoked when a socket changes state. Callback signature: `sock_state_cb(self, fd, readable, writable)`
- **socket_send_buffer_size** (*int*) – Size for the created socket's send buffer.
- **socket_receive_buffer_size** (*int*) – Size for the created socket's receive buffer.
- **rotate** (*bool*) – If set to True, the nameservers are rotated when doing queries.

The c-ares `Channel` provides asynchronous DNS operations.

gethostbyname (*name*, *family*, *callback*)

Parameters

- **name** (*string*) – Name to query.
- **family** (*int*) – Socket family.
- **callback** (*callable*) – Callback to be called with the result of the query.

Retrieves host information corresponding to a host name from a host database.

Callback signature: `callback(result, errorno)`

gethostbyaddr (*name*, *callback*)

Parameters

- **name** (*string*) – Name to query.
- **callback** (*callable*) – Callback to be called with the result of the query.

Retrieves the host information corresponding to a network address.

Callback signature: `callback(result, errorno)`

getnameinfo (*name*, *port*, *flags*, *callback*)

Parameters

- **name** (*string*) – Name to query.
- **port** (*int*) – Port of the service to query.
- **flags** (*int*) – Query flags, see the NI flags section.
- **callback** (*callable*) – Callback to be called with the result of the query.

Provides protocol-independent name resolution from an address to a host name and from a port number to the service name.

Callback signature: `callback(result, errorno)`

query (*name*, *query_type*, *callback*)

Parameters

- **name** (*string*) – Name to query.
- **query_type** (*int*) – Type of query to perform.

- **callback** (*callable*) – Callback to be called with the result of the query.

Do a DNS query of the specified type. Available types:

- QUERY_TYPE_A
- QUERY_TYPE_AAAA
- QUERY_TYPE_CNAME
- QUERY_TYPE_MX
- QUERY_TYPE_NAPTR
- QUERY_TYPE_NS
- QUERY_TYPE_PTR
- QUERY_TYPE_SOA
- QUERY_TYPE_SRV
- QUERY_TYPE_TXT

Callback signature: `callback(result, errorno)`. The result type varies depending on the query type:

- A and AAAA: `ares_query_simple_result`, fields:
  - host
  - ttl
- CNAME: `ares_query_cname_result`, fields:
  - cname
  - ttl
- MX: `ares_query_mx_result`, fields:
  - host
  - priority
  - ttl
- NAPTR: `ares_query_naptr_result`, fields:
  - order
  - preference
  - flags
  - service
  - regex
  - replacement
  - ttl
- NS: `ares_query_ns_result`, fields:
  - host
  - ttl
- PTR: `ares_query_ptr_result`, fields:

  - name
  - ttl
- SOA: `ares_query_soa_result`, fields:
  - nsmane
  - hostmaster
  - serial
  - refresh
  - retry
  - expires
  - minttl
  - ttl
- SRV: `ares_query_srv_result`, fields:
  - host
  - port
  - priority
  - weight
  - ttl
- TXT: `ares_query_txt_result`, fields:
  - text
  - ttl

Note: TTL is not implemented for CNAME, NS and PTR), so it's set to None.

`cancel`()

Cancel any pending query on this channel. All pending callbacks will be called with ARES_ECANCELLED errorno.

`destroy`()

Destroy the channel. All pending callbacks will be called with ARES_EDESTRUCTION errorno.

`process_fd`(*read_fd*, *write_fd*)

Parameters

- **`read_fd`** (*`int`*) – File descriptor ready to read from.
- **`write_fd`** (*`int`*) – File descriptor ready to write to.

Process the given file descriptors for read and/or write events.

`getsock`()

Return a tuple containing 2 lists with the file descriptors ready to read and write.

`timeout`([*max_timeout*])

Parameters **`max_timeout`** (*`float`*) – Maximum timeout.

Determines the maximum time for which the caller should wait before invoking `process_fd` to process timeouts. If the `max_timeout` parameter is specified, it is stored on the channel and the appropriate value is then returned.

`set_local_ip`(*local_ip*)

Parameters **`local_ip`** (*`str`*) – IP address.

Set the local IPv4 or IPv6 address from which the queries will be sent.

`set_local_dev`(*local_dev*)

Parameters **`local_dev`** (*`str`*) – Network device name.

Set the local ethernet device from which the queries will be sent.

`servers`

List of nameservers to use for DNS queries.

Utility functions

`pycares.`**`reverse_address`**(*ip_adress*)

Parameters **`ip_address`** (*`string`*) – IP address to be reversed.

Returns the reversed representation of an IP address, usually used when doing PTR queries.

Example:

```
pycares.reverse_address('1.2.3.4')
'4.3.2.1.in-addr.arpa'

pycares.reverse_address('2a03:2880:10:cf01:face:b00c::')
'0.0.0.0.0.0.0.0.c.0.0.b.e.c.a.f.1.0.f.c.0.1.0.0.0.8.8.2.3.0.a.2.ip6.arpa'
```

c-ares library constants

Channel flags

`pycares.`**`ARES_FLAG_USEVC`**

`pycares.`**`ARES_FLAG_PRIMARY`**

`pycares.`**`ARES_FLAG_IGNTC`**

`pycares.`**`ARES_FLAG_NORECURSE`**

`pycares.`**`ARES_FLAG_STAYOPEN`**

`pycares.`**`ARES_FLAG_NOSEARCH`**

`pycares.`**`ARES_FLAG_NOALIASES`**

`pycares.`**`ARES_FLAG_NOCHECKRESP`**

See also:

c-ares documentation for ares_init

Nameinfo constants

pycares.**ARES_NI_NOFQDN**

pycares.**ARES_NI_NUMERICHOST**

pycares.**ARES_NI_NAMEREQD**

pycares.**ARES_NI_NUMERICSERV**

pycares.**ARES_NI_DGRAM**

pycares.**ARES_NI_TCP**

pycares.**ARES_NI_UDP**

pycares.**ARES_NI_SCTP**

pycares.**ARES_NI_DCCP**

pycares.**ARES_NI_NUMERICSCOPE**

pycares.**ARES_NI_LOOKUPHOST**

pycares.**ARES_NI_LOOKUPSERVICE**

pycares.**ARES_NI_IDN**

pycares.**ARES_NI_IDN_ALLOW_UNASSIGNED**

pycares.**ARES_NI_IDN_USE_STD3_ASCII_RULES**

See also:

c-ares documentation for ares_getnameinfo

Others

pycares.**ARES_SOCKET_BAD**

pycares.errno — Error constant definitions

This module contains the defined error constants from c-ares.

pycares.errno.**errorcode**

Mapping (code, string) with c-ares error codes.

pycares.errno.**strerror**(*errorno*)

Parameters **errorno** (*int*) – Error number.

Get the string representation of the given c-ares error number.

Event loop integration

pycares can be integrated in an already existing event loop without much trouble. The examples folder contains several examples:

- cares-select.py: ntegration with plain select
- cares-resolver.py: integration with the pyuv event loop
- cares-asyncio.py: integration with the asyncio framework

Additionally, Tornado provides integration with pycaes through a resolver module.

CHAPTER 2

ToDo

2.1 Things yet to be done

```
- channel init options: sortlist
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index