## NAME

ares_set_pending_write_cb, ares_process_pending_write − Function for setting a callback which is triggered when there is potential pending data which needs to be written.

## SYNOPSIS

```
#include <ares.h>

typedef void (*ares_pending_write_cb)(void *data);

void ares_set_pending_write_cb(
  ares_channel_t       *channel,
  ares_pending_write_cb  callback,
  void                 *user_data);

void ares_process_pending_write(ares_channel_t *channel);
```

## DESCRIPTION

The **ares_set_pending_write_cb(3)** function sets a callback function *callback* in the given ares channel handle *channel* that is invoked whenever there is new pending TCP data to be written. Since TCP is stream based, if there are multiple queries being enqueued back to back they can be sent as one large buffer. Normally a **send(2)** syscall operation would be triggered for each query.

When setting this callback, an event will be triggered when data is buffered, but not written. This event is used to wake the caller's event loop which should call **ares_process_pending_write(3)** using the channel associated with the callback. Each time the callback is triggered must result in a call to **ares_process_pending_write(3)** from the caller's event loop otherwise stalls and timeouts may occur. The callback **must not** call **ares_process_pending_write(3)** directly as otherwise it would invalidate any advantage of this use-case.

This is considered an optimization, especially when using TLS-based connections which add additional overhead to the data stream. Due to the asyncronous nature of c-ares, there is no way to identify when a caller may be finished enqueuing queries via any of the possible public API calls such as **ares_getaddrinfo(3)** or **ares_search_dnsrec(3)**, so this is an enhancement to try to group query send operations together and will rely on the singaling latency involved in waking the user's event loop.

If no callback is set, data will be written immediately to the socket, thus bypassing this optimization.

This option cannot be used with *ARES_OPT_EVENT_THREAD* passed to **ares_init_options(3)** since the user has no event loop. This optimization is automatically enabled when using the Event Thread as it sets the callback for its own internal signaling.

## AVAILABILITY

This function was first introduced in c-ares version 1.34.0.

## SEE ALSO

**ares_init_options**(3)