NAME

ares_fds − return file descriptors to select on (deprecated)

SYNOPSIS

#include <ares.h>

int ares_fds(const ares_channel_t **channel*,
fd_set **read_fds*,
fd_set **write_fds*)

DESCRIPTION

See the **NOTES** section on issues with this function and alternatives.

The **ares_fds(3)** function retrieves the set of file descriptors which the calling application should **select(2)** on for reading and writing for the processing of name service queries pending on the name service channel identified by *channel*. Should not be used with**ARES_OPT_EVENT_THREAD** is passed to *ares_init_options(3)*.

File descriptors will be set in the file descriptor sets pointed to by *read_fds* and *write_fds* as appropriate. File descriptors already set in *read_fds* and *write_fds* will remain set; initialization of the file descriptor sets (using **FD_ZERO**) is the responsibility of the caller.

RETURN VALUES

ares_fds(3) returns a value that is one greater than the number of the highest socket set in either *read_fds* or *write_fds*. If no queries are active, **ares_fds(3)** returns 0.

NOTES

The **select(2)** call which takes the *fd_set* parameter has significant limitations which can impact modern systems. The limitations can vary from system to system, but in general if the file descriptor value itself is greater than 1024 (not the count but the actual value), this can lead to **ares_fds(3)** writing out of bounds which will cause a system crash. In modern networking clients, it is not unusual to have file descriptor values above 1024, especially when a library is pulled in as a dependency into a larger project.

c-ares does not attempt to detect this condition to prevent crashes due to both implementation-defined behavior in the OS as well as integrator-controllable tunables which may impact the limits.

It is recommended to use **ARES_OPT_EVENT_THREAD** passed to *ares_init_options(3)*, or socket state callbacks (**ARES_OPT_SOCK_STATE_CB**) registered via *ares_init_options(3)* and use more modern methods to check for socket readable/writable state such as *poll(2)*, *epoll(2)*, or *kqueue(2)*.

SEE ALSO

ares_init_options(3), **ares_timeout**(3), **ares_process**(3)