NAME

ares_library_init − c-ares library initialization

SYNOPSIS

```
#include <ares.h>

int ares_library_init(int flags)

int ares_library_init_mem(int flags,
                          void *(*amalloc)(size_t),
                          void (*afree)(void *ptr),
                          void (*arealloc)(void *ptr, size_t size))
```

DESCRIPTION

The **ares_library_init** function performs initializations internally required by the c-ares library that must take place before any other function provided by c-ares can be used in a program.

This function must be called at least once within the life of a program, before the program actually executes any other c-ares library function. Initializations done by this function remain effective until a number of calls to *ares_library_cleanup(3)* equal to the number of calls to this function are performed.

Successive calls to this function do nothing further, only the first call done when c-ares is in an uninitialized state is actually effective.

The *flags* parameter is a bit pattern that tells c-ares exactly which features should be initialized, as described below. Set the desired bits by ORing the values together. In normal operation you should specify *ARES_LIB_INIT_ALL*. Don't use any other value unless you are familiar with it and trying to control some internal c-ares feature.

The **ares_library_init_mem** function allows the caller to provide memory management functions that the c-ares library will be use instead of *malloc(3)*, *free(3)* and *realloc(3)*.

This function is not thread safe. You have to call it once the program has started, but this call must be done before the program starts any other thread. This is required to avoid potential race conditions in library initialization, and also due to the fact that *ares_library_init(3)* might call functions from other libraries that are thread unsafe, and could conflict with any other thread that is already using these other libraries.

Win32/64 application DLLs shall not call *ares_library_init(3)* from the DllMain function. Doing so will produce deadlocks and other problems.

FLAGS

ARES_LIB_INIT_ALL
: Initialize everything possible. This sets all known bits.

ARES_LIB_INIT_WIN32
: Initialize Win32/64 specific libraries.

ARES_LIB_INIT_NONE
: Initialize nothing extra. This sets no bit.

RETURN VALUE

Upon successful completion, ares_library_init() will return 0. Otherwise, a non-zero error number will be returned to indicate the error. Except for *ares_strerror(3)*, you shall not call any other c-ares function upon *ares_library_init(3)* failure.

AVAILABILITY

This function was first introduced in c-ares version 1.7.0 along with the definition of preprocessor symbol *CARES_HAVE_ARES_LIBRARY_INIT* as an indication of the availability of this function. Its recursive behavior, which requires a matching number of calls to *ares_library_cleanup()* in order to deinitialize the library, is present since c-ares version 1.10.0. Earlier versions would deinitialize the library on the first call to *ares_library_cleanup()*.

Since the introduction of this function it is absolutely mandatory to call it for any Win32/64 program using c-ares.

Non-Win32/64 systems can still use c-ares version 1.7.0 without calling *ares_library_init(3)* due to the fact that *currently* it is nearly a do-nothing function on non-Win32/64 platforms at this point.

SEE ALSO

ares_library_cleanup(3), ares_strerror(3)

AUTHOR

Yang Tse

Copyright 1998 by the Massachusetts Institute of Technology.
Copyright (C) 2004-2009 by Daniel Stenberg.