

NAME

curl_multi_wakeup – wake up a sleeping curl_multi_poll call

SYNOPSIS

```
#include <curl/curl.h>
```

```
CURLMcode curl_multi_wakeup(CURLM *multi_handle);
```

DESCRIPTION

This function can be called from any thread and it wakes up a sleeping *curl_multi_poll(3)* call that is currently (or is about to be) waiting for activity or a timeout.

If the function is called when there is no *curl_multi_poll(3)* call, it causes the next call to return immediately.

Calling this function only guarantees to wake up the current (or the next if there is no current) *curl_multi_poll(3)* call, which means it is possible that multiple calls to this function wake up the same waiting operation.

This function has no effect on *curl_multi_wait(3)* calls.

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
extern int time_to_die(void);
extern int set_something_to_signal_thread_1_to_exit(void);
extern int decide_to_stop_thread1();

int main(void)
{
    CURL *easy;
    CURLM *multi = curl_multi_init();
    int still_running;

    easy = curl_easy_init();

    /* add the individual easy handle */
    curl_multi_add_handle(multi, easy);

    /* this is thread 1 */
    do {
        CURLMcode mresult;
        int numfds;

        mresult = curl_multi_perform(multi, &still_running);

        if(mresult == CURLM_OK) {
            /* wait for activity, timeout or wakeup */
            mresult = curl_multi_poll(multi, NULL, 0, 10000, &numfds);
        }

        if(time_to_die())
            return 1;

    } while(still_running);
```

```
curl_multi_remove_handle(multi, easy);

/* this is thread 2 */

if(decide_to_stop_thread1()) {

    set_something_to_signal_thread_1_to_exit();

    curl_multi_wakeup(multi);
}
}
```

AVAILABILITY

Added in curl 7.68.0

RETURN VALUE

This function returns a CURLMcode indicating success or error.

CURLM_OK (0) means everything was OK, non-zero means an error occurred, see *libcurl-errors(3)*.

SEE ALSO

curl_multi_poll(3), curl_multi_wait(3)