NAME

curl_multi_timeout − how long to wait for action before proceeding

SYNOPSIS

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

CURLMcode curl_multi_timeout(CURLM *multi_handle, long *timeout);
```

DESCRIPTION

An application using the libcurl multi interface should call *curl_multi_timeout(3)* to figure out how long it should wait for socket actions − at most − before proceeding.

Proceeding means either doing the socket−style timeout action: call the *curl_multi_socket_action(3)* function with the **sockfd** argument set to CURL_SOCKET_TIMEOUT, or call *curl_multi_perform(3)* if you are using the simpler and older multi interface approach.

The timeout value returned in the long **timeout_ms** points to, is in number of milliseconds at this moment. If 0, it means you should proceed immediately without waiting for anything. If it returns −1, there is no timeout at all set.

An application that uses the *multi_socket* API should not use this function. It should instead use the *CURLMOPT_TIMERFUNCTION(3)* option for proper and desired behavior.

Note: if libcurl returns a −1 timeout here, it means that libcurl currently has no stored timeout value. You must not wait too long (more than a few seconds perhaps) before you call *curl_multi_perform(3)* again.

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
int main(void)
{
 struct timeval timeout;
 long timeo;
 fd_set fdread;
 fd_set fdwrite;
 fd_set fdexcep;
 int maxfd = 2;
 CURLM *multi = curl_multi_init();

 curl_multi_timeout(multi, &timeo);
 if(timeo < 0)
  /* no set timeout, use a default */
  timeo = 980;

 timeout.tv_sec = timeo / 1000;
 timeout.tv_usec = (timeo % 1000) * 1000;

 /* wait for activities no longer than the set timeout */
 select(maxfd + 1, &fdread, &fdwrite, &fdexcep, &timeout);
}
```

TYPICAL USAGE

Call *curl_multi_timeout(3)*, then wait for action on the sockets. Figure out which sockets to wait for by calling *curl_multi_fdset(3)*.

When there is activity or timeout, call *curl_multi_perform(3)* and then loop − until all transfers are

complete.

AVAILABILITY

Added in curl 7.15.4

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_fdset(3), **curl_multi_info_read**(3), **curl_multi_setopt**(3), **curl_multi_socket**(3)