

**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're using the simpler and older multi interface approach.

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

An application that uses the multi\_socket API SHOULD not use this function, but SHOULD instead use *curl\_multi\_setopt(3)* and its *CURLMOPT\_TIMERFUNCTION* option for proper and desired behavior.

Note: if libcurl returns a -1 timeout here, it just 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() again.

**RETURN VALUE**

The standard CURLMcode for multi interface error codes.

**TYPICAL USAGE**

Call **curl\_multi\_timeout(3)**, then wait for action on the sockets. You figure out which sockets to wait for by calling **curl\_multi\_fdset(3)** or by a previous call to **curl\_multi\_socket(3)**.

**AVAILABILITY**

This function was added in libcurl 7.15.4.

**SEE ALSO**

**curl\_multi\_fdset(3)**, **curl\_multi\_info\_read(3)**, **curl\_multi\_socket(3)**, **curl\_multi\_setopt(3)**