NAME
>   CURLMOPT_TIMERFUNCTION − set callback to receive timeout values

SYNOPSIS
>   #include <curl/curl.h>
>
>   int timer_callback(CURLM *multi,   /* multi handle */
>   long timeout_ms, /* see above */
>   void *userp);   /* private callback pointer */
>
>   CURLMcode curl_multi_setopt(CURLM *handle, CURLMOPT_TIMERFUNCTION, timer_callback);

DESCRIPTION
>   Pass a pointer to your callback function, which should match the prototype shown above.
>
>   Certain features, such as timeouts and retries, require you to call libcurl even when there is no activity on the file descriptors.
>
>   Your callback function **timer_callback** should install a non-repeating timer with an interval of **timeout_ms**. Each time that timer fires, call either *curl_multi_socket_action(3)* or *curl_multi_perform(3)*, depending on which interface you use.
>
>   A **timeout_ms** value of -1 means you should delete your timer.
>
>   A **timeout_ms** value of 0 means you should call *curl_multi_socket_action(3)* or *curl_multi_perform(3)* (once) as soon as possible.
>
>   **timer_callback** will only be called when the **timeout_ms** changes.
>
>   The **userp** pointer is set with *CURLMOPT_TIMERDATA(3)*.
>
>   The timer callback should return 0 on success, and -1 on error. This callback can be used instead of, or in addition to, *curl_multi_timeout(3)*.

DEFAULT
>   NULL

PROTOCOLS
>   All

EXAMPLE
```
static gboolean timeout_cb(gpointer user_data) {
  if (user_data) {
    g_free(user_data);
    curl_multi_setopt(curl_handle, CURLMOPT_TIMERDATA, NULL);
  }
  int running;
  curl_multi_socket_action(multi, CURL_SOCKET_TIMEOUT, 0, &running);
  return G_SOURCE_REMOVE;
}

static int timerfunc(CURLM *multi, long timeout_ms, void *userp) {
  guint *id = userp;

  if (id)
    g_source_remove(*id);
```

```
      // -1 means we should just delete our timer.
      if (timeout_ms == -1) {
        g_free(id);
        id = NULL;
      } else {
        if (!id)
          id = g_new(guint, 1);
        *id = g_timeout_add(timeout_ms, timeout_cb, id);
      }
      curl_multi_setopt(multi, CURLMOPT_TIMERDATA, id);
      return 0;
    }

    curl_multi_setopt(multi, CURLMOPT_TIMERFUNCTION, timerfunc);
```

AVAILABILITY

Added in 7.16.0

RETURN VALUE

Returns CURLM_OK if the option is supported, and CURLM_UNKNOWN_OPTION if not.

SEE ALSO

CURLMOPT_TIMERDATA(3), **CURLMOPT_SOCKETFUNCTION**(3),