NAME

CURLSHOPT_LOCKFUNC − mutex lock callback

SYNOPSIS

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

void lockcb(CURL *handle, curl_lock_data data, curl_lock_access access,
      void *clientp);

CURLSHcode curl_share_setopt(CURLSH *share, CURLSHOPT_LOCKFUNC, lockcb);
```

DESCRIPTION

Set a mutex lock callback for the share object, to allow it to get used by multiple threads concurrently. There is a corresponding *CURLSHOPT_UNLOCKFUNC(3)* callback called when the mutex is again released.

The *lockcb* argument must be a pointer to a function matching the prototype shown above. The arguments to the callback are:

handle is the currently active easy handle in use when the share object is intended to get used.

The *data* argument tells what kind of data libcurl wants to lock. Make sure that the callback uses a different lock for each kind of data.

access defines what access type libcurl wants, shared or single.

clientp is the private pointer you set with *CURLSHOPT_USERDATA(3)*. This pointer is not used by libcurl itself.

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
extern void mutex_lock(CURL *handle, curl_lock_data data,
                       curl_lock_access access, void *clientp);

int main(void)
{
 CURLSHcode sh;
 CURLSH *share = curl_share_init();
 sh = curl_share_setopt(share, CURLSHOPT_LOCKFUNC, mutex_lock);
 if(sh)
   printf("Error: %s\n", curl_share_strerror(sh));
}
```

AVAILABILITY

Added in curl 7.10.3

RETURN VALUE

CURLSHE_OK (zero) means that the option was set properly, non−zero means an error occurred. See *libcurl−errors(3)* for the full list with descriptions.

SEE ALSO

CURLSHOPT_UNLOCKFUNC(3), **curl_share_cleanup**(3), **curl_share_init**(3), **curl_share_setopt**(3)