

**NAME**

libcurl-share – how to use the share interface

**DESCRIPTION**

This is an overview on how to use the libcurl share interface in your C programs. There are specific man pages for each function mentioned in here.

All functions in the share interface are prefixed with `curl_share`.

**OBJECTIVES**

The share interface was added to enable sharing of data between curl "handles".

**ONE SET OF DATA - MANY TRANSFERS**

You can have multiple easy handles share data between them. Have them update and use the **same** cookie database or DNS cache! This way, each single transfer will take advantage from data updates made by the other transfer(s).

**SHARE OBJECT**

You create a shared object with `curl_share_init(3)`. It returns a handle for a newly created one.

You tell the shared object what data you want it to share by using `curl_share_setopt(3)`.

Since you can use this share from multiple threads, and libcurl has no internal thread synchronization, you must provide mutex callbacks if you're using this multi-threaded. You set lock and unlock functions with `curl_share_setopt(3)` too.

Then, you make an easy handle to use this share, you set the `CURLOPT_SHARE` option with `curl_easy_setopt(3)`, and pass in share handle. You can make any number of easy handles share the same share handle.

To make an easy handle stop using that particular share, you set `CURLOPT_SHARE` to `NULL` for that easy handle. To make a handle stop sharing a particular data, you can `CURLSHOPT_UNSHARE` it.

When you're done using the share, make sure that no easy handle is still using it, and call `curl_share_cleanup(3)` on the handle.

**SEE ALSO**

`curl_share_init(3)`, `curl_share_setopt(3)`, `curl_share_cleanup(3)`