

NAME

CURLOPT_COOKIELIST – add to or manipulate cookies held in memory

SYNOPSIS

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

```
CURLcode curl_easy_setopt(CURL *handle, CURLOPT_COOKIELIST,
                           char *cookie);
```

DESCRIPTION

Pass a char * to a *cookie* string.

Such a cookie can be either a single line in Netscape / Mozilla format or just regular HTTP-style header (Set-Cookie: ...) format. This will also enable the cookie engine. This adds that single cookie to the internal cookie store.

If you use the Set-Cookie format and don't specify a domain then the cookie is sent for any domain and will not be modified. If a server sets a cookie of the same name (or maybe you've imported one) then both will be sent on a future transfer to that server, likely not what you intended. Either set a domain in Set-Cookie (doing that will include sub domains) or use the Netscape format as shown in EXAMPLE.

Starting in 7.43.0 the aforementioned any-domain cookies will not appear in the lists exported by *CURLINFO_COOKIELIST(3)* and *CURLOPT_COOKIEJAR(3)*.

Additionally, there are commands available that perform actions if you pass in these exact strings:

ALL erases all cookies held in memory

SESS erases all session cookies held in memory

FLUSH

writes all known cookies to the file specified by *CURLOPT_COOKIEJAR(3)*

RELOAD

loads all cookies from the files specified by *CURLOPT_COOKIEFILE(3)*

DEFAULT

NULL

PROTOCOLS

HTTP

EXAMPLE

```
/* This example shows an inline import of a cookie in Netscape format.
You can set the cookie as HttpOnly to prevent XSS attacks by prepending
#HttpOnly_ to the hostname. That may be useful if the cookie will later
be imported by a browser.
*/
```

```
#define SEP "\t" /* Tab separates the fields */
```

```
char *my_cookie =
"example.com" /* Hostname */
SEP "FALSE" /* Include subdomains */
SEP "/" /* Path */
SEP "FALSE" /* Secure */
```

```
SEP "0"      /* Expiry in epoch time format. 0 == Session */
SEP "foo"    /* Name */
SEP "bar";   /* Value */

/* my_cookie is imported immediately via CURLOPT_COOKIELIST.
*/
curl_easy_setopt(curl, CURLOPT_COOKIELIST, my_cookie);

/* The list of cookies in cookies.txt will not be imported until right
before a transfer is performed. Cookies in the list that have the same
hostname, path and name as in my_cookie are skipped. That is because
libcurl has already imported my_cookie and it's considered a "live"
cookie. A live cookie won't be replaced by one read from a file.
*/
curl_easy_setopt(curl, CURLOPT_COOKIEFILE, "cookies.txt"); /* import */

/* Cookies are exported after curl_easy_cleanup is called. The server
may have added, deleted or modified cookies by then. The cookies that
were skipped on import are not exported.
*/
curl_easy_setopt(curl, CURLOPT_COOKIEJAR, "cookies.txt"); /* export */

res = curl_easy_perform(curl); /* cookies imported from cookies.txt */

curl_easy_cleanup(curl); /* cookies exported to cookies.txt */
```

AVAILABILITY

ALL was added in 7.14.1

SESS was added in 7.15.4

FLUSH was added in 7.17.1

RELOAD was added in 7.39.0

RETURN VALUE

Returns `CURLE_OK` if the option is supported, `CURLE_UNKNOWN_OPTION` if not, or `CURLE_OUT_OF_MEMORY` if there was insufficient heap space.

SEE ALSO

CURLOPT_COOKIEFILE(3), CURLOPT_COOKIEJAR(3), CURLOPT_COOKIE(3),