

**NAME**

curl\_easy\_getinfo - extract information from a curl handle

**SYNOPSIS**

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

```
CURLcode curl_easy_getinfo(CURL *curl, CURLINFO info, ... );
```

**DESCRIPTION**

Request internal information from the curl session with this function. The third argument **MUST** be a pointer to a long, a pointer to a char \*, a pointer to a struct curl\_slist \* or a pointer to a double (as this documentation describes further down). The data pointed-to will be filled in accordingly and can be relied upon only if the function returns CURLE\_OK. Use this function **AFTER** a performed transfer if you want to get transfer- oriented data.

You should not free the memory returned by this function unless it is explicitly mentioned below.

**AVAILABLE INFORMATION**

The following information can be extracted:

**CURLINFO\_EFFECTIVE\_URL**

Pass a pointer to a 'char \*' to receive the last used effective URL.

**CURLINFO\_RESPONSE\_CODE**

Pass a pointer to a long to receive the last received HTTP or FTP code. This option was known as **CURLINFO\_HTTP\_CODE** in libcurl 7.10.7 and earlier. This will be zero if no server response code has been received. Note that a proxy's **CONNECT** response should be read with **CURLINFO\_HTTP\_CONNECTCODE** and not this.

**CURLINFO\_HTTP\_CONNECTCODE**

Pass a pointer to a long to receive the last received proxy response code to a **CONNECT** request.

**CURLINFO\_FILETIME**

Pass a pointer to a long to receive the remote time of the retrieved document (in number of seconds since 1 jan 1970 in the GMT/UTC time zone). If you get -1, it can be because of many reasons (unknown, the server hides it or the server doesn't support the command that tells document time etc) and the time of the document is unknown. Note that you must tell the server to collect this information before the transfer is made, by using the **CURLOPT\_FILETIME** option to *curl\_easy\_setopt(3)* or you will unconditionally get a -1 back. (Added in 7.5)

**CURLINFO\_TOTAL\_TIME**

Pass a pointer to a double to receive the total time in seconds for the previous transfer, including name resolving, TCP connect etc.

**CURLINFO\_NAMELOOKUP\_TIME**

Pass a pointer to a double to receive the time, in seconds, it took from the start until the name resolving was completed.

**CURLINFO\_CONNECT\_TIME**

Pass a pointer to a double to receive the time, in seconds, it took from the start until the connect to the remote host (or proxy) was completed.

**CURLINFO\_PRETRANSFER\_TIME**

Pass a pointer to a double to receive the time, in seconds, it took from the start until the file transfer is just about to begin. This includes all pre-transfer commands and negotiations that are specific to the particular protocol(s) involved.

**CURLINFO\_STARTTRANSFER\_TIME**

Pass a pointer to a double to receive the time, in seconds, it took from the start until the first byte is just about to be transferred. This includes **CURLINFO\_PRETRANSFER\_TIME** and also the time the server needs to calculate the result.

**CURLINFO\_REDIRECT\_TIME**

Pass a pointer to a double to receive the total time, in seconds, it took for all redirection steps include name lookup, connect, pretransfer and transfer before final transaction was started. **CURLINFO\_REDIRECT\_TIME** contains the complete execution time for multiple redirections. (Added in 7.9.7)

**CURLINFO\_REDIRECT\_COUNT**

Pass a pointer to a long to receive the total number of redirections that were actually followed. (Added in 7.9.7)

**CURLINFO\_SIZE\_UPLOAD**

Pass a pointer to a double to receive the total amount of bytes that were uploaded.

**CURLINFO\_SIZE\_DOWNLOAD**

Pass a pointer to a double to receive the total amount of bytes that were downloaded. The amount is only for the latest transfer and will be reset again for each new transfer.

**CURLINFO\_SPEED\_DOWNLOAD**

Pass a pointer to a double to receive the average download speed that curl measured for the complete download. Measured in bytes/second.

**CURLINFO\_SPEED\_UPLOAD**

Pass a pointer to a double to receive the average upload speed that curl measured for the complete upload. Measured in bytes/second.

**CURLINFO\_HEADER\_SIZE**

Pass a pointer to a long to receive the total size of all the headers received. Measured in number of bytes.

**CURLINFO\_REQUEST\_SIZE**

Pass a pointer to a long to receive the total size of the issued requests. This is so far only for HTTP requests. Note that this may be more than one request if **FOLLOWLOCATION** is true.

**CURLINFO\_SSL\_VERIFYRESULT**

Pass a pointer to a long to receive the result of the certification verification that was requested (using the **CURLOPT\_SSL\_VERIFYPEER** option to *curl\_easy\_setopt(3)*).

**CURLINFO\_SSL\_ENGINES**

Pass the address of a 'struct curl\_slist \*' to receive a linked-list of OpenSSL crypto-engines supported. Note that engines are normally implemented in separate dynamic libraries. Hence not all the returned engines may be available at run-time. **NOTE:** you must call *curl\_slist\_free\_all(3)* on the list pointer once you're done with it, as libcurl will not free the data for you. (Added in 7.12.3)

**CURLINFO\_CONTENT\_LENGTH\_DOWNLOAD**

Pass a pointer to a double to receive the content-length of the download. This is the value read from the Content-Length: field.

**CURLINFO\_CONTENT\_LENGTH\_UPLOAD**

Pass a pointer to a double to receive the specified size of the upload.

**CURLINFO\_CONTENT\_TYPE**

Pass a pointer to a 'char \*' to receive the content-type of the downloaded object. This is the value read from the Content-Type: field. If you get **NULL**, it means that the server didn't send a valid Content-Type header or that the protocol used doesn't support this.

**CURLINFO\_PRIVATE**

Pass a pointer to a 'char \*' to receive the pointer to the private data associated with the curl handle (set with the **CURLOPT\_PRIVATE** option to *curl\_easy\_setopt(3)*). Please note that for internal reasons, the value is returned as a 'char \*', although effectively being a 'void \*'. (Added in 7.10.3)

**CURLINFO\_HTTPAUTH\_AVAIL**

Pass a pointer to a long to receive a bitmask indicating the authentication method(s) available. The meaning of the bits is explained in the `CURLOPT_HTTPAUTH` option for `curl_easy_setopt(3)`. (Added in 7.10.8)

**CURLINFO\_PROXYAUTH\_AVAIL**

Pass a pointer to a long to receive a bitmask indicating the authentication method(s) available for your proxy authentication. (Added in 7.10.8)

**CURLINFO\_OS\_ERRNO**

Pass a pointer to a long to receive the errno variable from a connect failure. (Added in 7.12.2)

**CURLINFO\_NUM\_CONNECTS**

Pass a pointer to a long to receive how many new connections libcurl had to create to achieve the previous transfer (only the successful connects are counted). Combined with `CURLINFO_REDIRECT_COUNT` you are able to know how many times libcurl successfully reused existing connection(s) or not. See the Connection Options of `curl_easy_setopt(3)` to see how libcurl tries to make persistent connections to save time. (Added in 7.12.3)

**CURLINFO\_COOKIELIST**

Pass a pointer to a 'struct curl\_slist \*' to receive a linked-list of all cookies cURL knows (expired ones, too). Don't forget to `curl_slist_free_all(3)` the list after it has been used. If there are no cookies (cookies for the handle have not been enabled or simply none have been received) 'struct curl\_slist \*' will be set to point to NULL. (Added in 7.14.1)

**CURLINFO\_LASTSOCKET**

Pass a pointer to a long to receive the last socket used by this curl session. If the socket is no longer valid, -1 is returned. When you finish working with the socket, you must call `curl_easy_cleanup()` as usual and let libcurl close the socket and cleanup other resources associated with the handle. This is typically used in combination with `CURLOPT_CONNECT_ONLY`. (Added in 7.15.2)

**CURLINFO\_FTP\_ENTRY\_PATH**

Pass a pointer to a 'char \*' to receive a pointer to a string holding the path of the entry path. That is the initial path libcurl ended up in when logging on to the remote FTP server. This stores a NULL as pointer if something is wrong. (Added in 7.15.4)

**TIMES**

An overview of the six time values available from `curl_easy_getinfo()`

**curl\_easy\_perform()**

```
|
|--NT
|--|--CT
|--|--|--PT
|--|--|--|--ST
|--|--|--|--|--TT
|--|--|--|--|--|--RT
```

- NT** `CURLINFO_NAMELOOKUP_TIME`. The time it took from the start until the name resolving was completed.
- CT** `CURLINFO_CONNECT_TIME`. The time it took from the start until the connect to the remote host (or proxy) was completed.
- PT** `CURLINFO_PRETRANSFER_TIME`. The time it took from the start until the file transfer is just about to begin. This includes all pre-transfer commands and negotiations that are specific to the particular protocol(s) involved.
- ST** `CURLINFO_STARTTRANSFER_TIME`. The time it took from the start until the first byte is just about to be transferred.

TT *CURLINFO\_TOTAL\_TIME*. Total time of the previous request.

RT *CURLINFO\_REDIRECT\_TIME*. The time it took for all redirection steps include name lookup, connect, pretransfer and transfer before final transaction was started. So, this is zero if no redirection took place.

#### **RETURN VALUE**

If the operation was successful, CURLE\_OK is returned. Otherwise an appropriate error code will be returned.

#### **SEE ALSO**

**curl\_easy\_setopt(3)**