NAME

CURLOPT_CUSTOMREQUEST − custom request method

SYNOPSIS

#include <curl/curl.h>

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_CUSTOMREQUEST, char *method);

DESCRIPTION

Pass a pointer to a null−terminated string as parameter.

When changing the request *method* by setting *CURLOPT_CUSTOMREQUEST(3)*, you do not actually change how libcurl behaves or acts: you only change the actual string sent in the request.

libcurl passes on the verbatim string in its request without any filter or other safe guards. That includes white space and control characters.

The application does not have to keep the string around after setting this option.

Using this option multiple times makes the last set string override the previous ones. Restore to the internal default by setting this to NULL.

This option can be used to specify the request:

HTTP Instead of GET or HEAD when performing HTTP based requests. This is particularly useful, for example, for performing an HTTP DELETE request.

For example:

When you tell libcurl to do a HEAD request, but then specify a GET though a custom request libcurl still acts as if it sent a HEAD. To switch to a proper HEAD use *CURLOPT_NOBODY(3)*, to switch to a proper POST use *CURLOPT_POST(3)* or *CURLOPT_POSTFIELDS(3)* and to switch to a proper GET use *CURLOPT_HTTPGET(3)*.

Many people have wrongly used this option to replace the entire request with their own, including multiple headers and POST contents. While that might work in many cases, it might cause libcurl to send invalid requests and it could possibly confuse the remote server badly. Use *CURLOPT_POST(3)* and *CURLOPT_POSTFIELDS(3)* to set POST data. Use *CURLOPT_HTTPHEADER(3)* to replace or extend the set of headers sent by libcurl. Use *CURLOPT_HTTP_VERSION(3)* to change the HTTP version.

When this option is used together with *CURLOPT_FOLLOWLOCATION(3)*, the custom set method overrides the method libcurl could otherwise change to for the subsequent requests. You can fine−tune that decision by using the CURLFOLLOW_OBEYCODE bit to *CURLOPT_FOLLOWLOCATION(3)* to make redirects adhere to the redirect response code as the protocol instructs.

FTP Instead of LIST and NLST when performing FTP directory listings.

IMAP Instead of LIST when issuing IMAP based requests.

POP3 Instead of LIST and RETR when issuing POP3 based requests.

For example:

When you tell libcurl to use a custom request it behaves like a LIST or RETR command was sent where it expects data to be returned by the server. As such *CURLOPT_NOBODY(3)* should be used when specifying commands such as **DELE** and **NOOP** for example.

SMTP Instead of a **HELP** or **VRFY** when issuing SMTP based requests.

For example:

Normally a multi line response is returned which can be used, in conjunction with *CURLOPT_MAIL_RCPT(3)*, to specify an EXPN request. If the *CURLOPT_NOBODY(3)* option is specified then the request can be used to issue **NOOP** and **RSET** commands.

DEFAULT

NULL

PROTOCOLS

This functionality affects ftp, http, imap, pop3 and smtp

EXAMPLE

```
int main(void)
{
  CURL *curl = curl_easy_init();
  if(curl) {
    CURLcode result;
    curl_easy_setopt(curl, CURLOPT_URL, "https://example.com/foo.bin");

    /* DELETE the given path */
    curl_easy_setopt(curl, CURLOPT_CUSTOMREQUEST, "DELETE");

    result = curl_easy_perform(curl);

    curl_easy_cleanup(curl);
  }
}
```

AVAILABILITY

Added in curl 7.1

RETURN VALUE

curl_easy_setopt(3) returns a CURLcode indicating success or error.

CURLE_OK (0) means everything was OK, non−zero means an error occurred, see *libcurl−errors(3)*.

SEE ALSO

CURLINFO_EFFECTIVE_METHOD(3), **CURLOPT_HTTPHEADER**(3), **CURLOPT_NOBODY**(3), **CURLOPT_REQUEST_TARGET**(3)