NAME

CURLOPT_UPLOAD − data upload

SYNOPSIS

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

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_UPLOAD, long upload);
```

DESCRIPTION

The long parameter *upload* set to 1 tells the library to prepare for and perform an upload. The *CURLOPT_READDATA(3)* and *CURLOPT_INFILESIZE(3)* or *CURLOPT_INFILESIZE_LARGE(3)* options are also interesting for uploads. If the protocol is HTTP, uploading means using the PUT request unless you tell libcurl otherwise.

Using PUT with HTTP 1.1 implies the use of a "Expect: 100−continue" header. You can disable this header with *CURLOPT_HTTPHEADER(3)* as usual.

If you use PUT to an HTTP 1.1 server, you can upload data without knowing the size before starting the transfer. The library enables this by adding a header "Transfer−Encoding: chunked". With HTTP 1.0 or if you prefer not to use chunked transfer, you must specify the size of the data with *CURLOPT_INFILESIZE(3)* or *CURLOPT_INFILESIZE_LARGE(3)*.

DEFAULT

0

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
static size_t read_cb(char *ptr, size_t size, size_t nmemb, void *userdata)
{
 FILE *src = userdata;
 /* copy as much data as possible into the 'ptr' buffer, but no more than
    'size' * 'nmemb' bytes */
 size_t retcode = fread(ptr, size, nmemb, src);

 return retcode;
}

int main(void)
{
 CURL *curl = curl_easy_init();
 if(curl) {
  CURLcode result;
  FILE *src = fopen("local-file", "r");
  curl_off_t fsize = 1234; /* set this to the size of the input file */

  /* we want to use our own read function */
  curl_easy_setopt(curl, CURLOPT_READFUNCTION, read_cb);

  /* enable uploading */
  curl_easy_setopt(curl, CURLOPT_UPLOAD, 1L);

  /* specify target */
  curl_easy_setopt(curl, CURLOPT_URL, "ftp://example.com/dir/to/newfile");

  /* now specify which pointer to pass to our callback */
```

```
    curl_easy_setopt(curl, CURLOPT_READDATA, src);

    /* Set the size of the file to upload */
    curl_easy_setopt(curl, CURLOPT_INFILESIZE_LARGE, (curl_off_t)fsize);

    /* Now run off and do what you have been told */
    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

CURLOPT_INFILESIZE_LARGE(3), **CURLOPT_PUT**(3), **CURLOPT_READFUNCTION**(3)