NAME

CURLOPT_POSTREDIR − how to act on a HTTP POST redirect

SYNOPSIS

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

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_POSTREDIR,
                          long bitmask);
```

DESCRIPTION

Pass a bitmask to control how libcurl acts on redirects after POSTs that get a 301, 302 or 303 response back. A parameter with bit 0 set (value **CURL_REDIR_POST_301**) tells the library to respect RFC2616/10.3.2 and not convert POST requests into GET requests when following a 301 redirection. Setting bit 1 (value **CURL_REDIR_POST_302**) makes libcurl maintain the request method after a 302 redirect whilst setting bit 2 (value **CURL_REDIR_POST_303**) makes libcurl maintain the request method after a 303 redirect. The value **CURL_REDIR_POST_ALL** is a convenience define that sets all three bits.

The non-RFC behaviour is ubiquitous in web browsers, so the library does the conversion by default to maintain consistency. However, a server may require a POST to remain a POST after such a redirection. This option is meaningful only when setting *CURLOPT_FOLLOWLOCATION(3)*.

DEFAULT

0

PROTOCOLS

HTTP(S)

EXAMPLE

```
CURL *curl = curl_easy_init();
if(curl) {
  curl_easy_setopt(curl, CURLOPT_URL, "http://example.com");

  /* a silly POST example */
  curl_easy_setopt(curl, CURLOPT_POSTFIELDS, "data=true");

  /* example.com is redirected, so we tell libcurl to send POST on 301, 302 and
     303 HTTP response codes */
  curl_easy_setopt(curl, CURLOPT_POSTREDIR, CURL_REDIR_POST_ALL);

  curl_easy_perform(curl);
}
```

AVAILABILITY

Added in 7.17.1. This option was known as CURLOPT_POST301 up to 7.19.0 as it only supported the 301 then. CURL_REDIR_POST_303 was added in 7.26.0.

RETURN VALUE

Returns CURLE_OK if the option is supported, and CURLE_UNKNOWN_OPTION if not.

SEE ALSO

CURLOPT_FOLLOWLOCATION(3), **CURLOPT_POSTFIELDS**(3),