NAME

error codes in libcurl

DESCRIPTION

This man page includes most, if not all, available error codes in libcurl. Why they occur and possibly what you can do to fix the problem.

CURLcode

Almost all "easy" interface functions return a CURLcode error code. No matter what, using *CURLOPT_ERRORBUFFER* is a good idea as it will give you a human readable error string that may offer more details about the error cause than just the error code does.

This man page is meant to describe libcurl 7.9.6 and later. Earlier versions might have had quirks not mentioned here.

CURLcode is one of the following:

CURLE_OK (0)
All fine. Proceed as usual.

CURLE_UNSUPPORTED_PROTOCOL (1)
The URL you passed to libcurl used a protocol that this libcurl does not support. The support might be a compile-time option that you didn't use, it can be a misspelled protocol string or just a protocol libcurl has no code for.

CURLE_FAILED_INIT (2)
Very early initialization code failed. This is likely to be an internal error or problem.

CURLE_URL_MALFORMAT (3)
The URL was not properly formatted.

CURLE_URL_MALFORMAT_USER (4)
URL user malformatted. The user-part of the URL syntax was not correct.

CURLE_COULDNT_RESOLVE_PROXY (5)
Couldn't resolve proxy. The given proxy host could not be resolved.

CURLE_COULDNT_RESOLVE_HOST (6)
Couldn't resolve host. The given remote host was not resolved.

CURLE_COULDNT_CONNECT (7)
Failed to connect() to host or proxy.

CURLE_FTP_WEIRD_SERVER_REPLY (8)
After connecting to a FTP server, libcurl expects to get a certain reply back. This error code implies that it god a strange or bad reply. The given remote server is probably not an OK FTP server.

CURLE_FTP_ACCESS_DENIED (9)
We were denied access when trying to login to an FTP server or when trying to change working directory to the one given in the URL.

CURLE_FTP_USER_PASSWORD_INCORRECT (10)
The username and/or the password were incorrect when trying to login to an FTP server.

CURLE_FTP_WEIRD_PASS_REPLY (11)
After having sent the FTP password to the server, libcurl expects a proper reply. This error code indicates that an unexpected code was returned.

CURLE_FTP_WEIRD_USER_REPLY (12)
After having sent user name to the FTP server, libcurl expects a proper reply. This error code indicates that an unexpected code was returned.

CURLE_FTP_WEIRD_PASV_REPLY (13)
libcurl failed to get a sensible result back from the server as a response to either a PASV or a EPSV command. The server is flawed.

CURLE_FTP_WEIRD_227_FORMAT (14)
FTP servers return a 227-line as a response to a PASV command. If libcurl fails to parse that line, this return code is passed back.

CURLE_FTP_CANT_GET_HOST (15)
An internal failure to lookup the host used for the new connection.

CURLE_FTP_CANT_RECONNECT (16)
A bad return code on either PASV or EPSV was sent by the FTP server, preventing libcurl from being able to continue.

CURLE_FTP_COULDNT_SET_BINARY (17)
Received an error when trying to set the transfer mode to binary.

CURLE_PARTIAL_FILE (18)
A file transfer was shorter or larger than expected. This happens when the server first reports an expected transfer size, and then delivers data that doesn't match the previously given size.

CURLE_FTP_COULDNT_RETR_FILE (19)
This was either a weird reply to a 'RETR' command or a zero byte transfer complete.

CURLE_FTP_WRITE_ERROR (20)
After a completed file transfer, the FTP server did not respond a proper

CURLE_FTP_QUOTE_ERROR (21)
When sending custom "QUOTE" commands to the remote server, one of the commands returned an error code that was 400 or higher.

CURLE_HTTP_RETURNED_ERROR (22)
This is returned if CURLOPT_FAILONERROR is set TRUE and the HTTP server returns an error code that is >= 400.

CURLE_WRITE_ERROR (23)
An error occurred when writing received data to a local file, or an error was returned to libcurl from a write callback.

CURLE_MALFORMAT_USER (24)
Malformat user. User name badly specified. *Not currently used*

CURLE_FTP_COULDNT_STOR_FILE (25)
FTP couldn't STOR file. The server denied the STOR operation. The error buffer usually contains the server's explanation to this.

CURLE_READ_ERROR (26)
There was a problem reading a local file or an error returned by the read callback.

CURLE_OUT_OF_MEMORY (27)
Out of memory. A memory allocation request failed. This is serious badness and things are severly screwed up if this ever occur.

CURLE_OPERATION_TIMEOUTED (28)
Operation timeout. The specified time-out period was reached according to the conditions.

CURLE_FTP_COULDNT_SET_ASCII (29)
libcurl failed to set ASCII transfer type (TYPE A).

CURLE_FTP_PORT_FAILED (30)
The FTP PORT command returned error. This mostly happen when you haven't specified a good enough address for libcurl to use. See *CURLOPT_FTPPORT*.

CURLE_FTP_COULDNT_USE_REST (31)
The FTP REST command returned error. This should never happen if the server is sane.

CURLE_FTP_COULDNT_GET_SIZE (32)
The FTP SIZE command returned errror. SIZE is not a kosher FTP command, it is an extension and not all servers support it. This is not a surprising error.

CURLE_HTTP_RANGE_ERROR (33)
The HTTP server does not support or accept range requests.

CURLE_HTTP_POST_ERROR (34)
This is an odd error that mainly occurs due to internal confusion.

CURLE_SSL_CONNECT_ERROR (35)
A problem occured somewhere in the SSL/TLS handshake. You really want the error buffer and read the message there as it pinpoints the problem slightly more. Could be certificates (file formats, paths, permissions), passwords, and others.

CURLE_FTP_BAD_DOWNLOAD_RESUME (36)
Attempting FTP resume beyond file size.

CURLE_FILE_COULDNT_READ_FILE (37)
A file given with FILE:// couldn't be opened. Most likely because the file path doesn't identify an existing file. Did you check file permissions?

CURLE_LDAP_CANNOT_BIND (38)
LDAP cannot bind. LDAP bind operation failed.

CURLE_LDAP_SEARCH_FAILED (39)
LDAP search failed.

CURLE_LIBRARY_NOT_FOUND (40)
Library not found. The LDAP library was not found.

CURLE_FUNCTION_NOT_FOUND (41)
Function not found. A required LDAP function was not found.

CURLE_ABORTED_BY_CALLBACK (42)
Aborted by callback. A callback returned "abort" to libcurl.

CURLE_BAD_FUNCTION_ARGUMENT (43)
Internal error. A function was called with a bad parameter.

CURLE_BAD_CALLING_ORDER (44)
Internal error. A function was called in a bad order.

CURLE_HTTP_PORT_FAILED (45)
Interface error. A specified outgoing interface could not be used. Set which interface to use for outgoing connections' source IP address with CURLOPT_INTERFACE.

CURLE_BAD_PASSWORD_ENTERED (46)
Bad password entered. An error was signaled when the password was entered. This can also be the result of a "bad password" returned from a specified password callback.

CURLE_TOO_MANY_REDIRECTS (47)
Too many redirects. When following redirects, libcurl hit the maximum amount. Set your limit with CURLOPT_MAXREDIRS.

CURLE_UNKNOWN_TELNET_OPTION (48)
An option set with CURLOPT_TELNETOPTIONS was not recognized/known. Refer to the appropriate documentation.

CURLE_TELNET_OPTION_SYNTAX (49)
A telnet option string was Illegally formatted.

CURLE_OBSOLETE (50)
This is not an error. This used to be another error code in an old libcurl version and is currently unused.

CURLE_SSL_PEER_CERTIFICATE (51)
The remote server's SSL certificate was deemed not OK.

CURLE_GOT_NOTHING (52)
Nothing was returned from the server, and under the circumstances, getting nothing is considered an error.

CURLE_SSL_ENGINE_NOTFOUND (53)
The specified crypto engine wasn't found.

CURLE_SSL_ENGINE_SETFAILED (54)
Failed setting the selected SSL crypto engine as default!

CURLE_SEND_ERROR (55)
Failed sending network data.

CURLE_RECV_ERROR (56)
Failure with receiving network data.

CURL_LAST
This is not an error, but in the curl/curl.h file this can be used to know how many existing error codes there are.

CURLMcode
This is the generic return code used by functions in the libcurl multi interface.