NAME

curl_easy_perform - perform a blocking file transfer

SYNOPSIS

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

CURLcode curl_easy_perform(CURL *easy_handle);
```

DESCRIPTION

curl_easy_perform(3) performs a network transfer in a blocking manner and returns when done, or earlier if it fails. For non-blocking behavior, see *curl_multi_perform(3)*.

Invoke this function after *curl_easy_init(3)* and all the *curl_easy_setopt(3)* calls are made, and it performs the transfer as described in the options. It must be called with the same **easy_handle** as input as the *curl_easy_init(3)* call returned.

You can do any amount of calls to *curl_easy_perform(3)* while using the same **easy_handle**. If you intend to transfer more than one file, you are even encouraged to do so. libcurl attempts to reuse existing connections for the following transfers, thus making the operations faster, less CPU intense and using less network resources. You probably want to use *curl_easy_setopt(3)* between the invokes to set options for the following *curl_easy_perform(3)* call.

You must never call this function simultaneously from two places using the same **easy_handle**. Let the function return first before invoking it another time. If you want parallel transfers, you must use several curl easy_handles.

A network transfer moves data to a peer or from a peer. An application tells libcurl how to receive data by setting the *CURLOPT_WRITEFUNCTION(3)* and *CURLOPT_WRITEDATA(3)* options. To tell libcurl what data to send, there are a few more alternatives but two common ones are *CURLOPT_READFUNCTION(3)* and *CURLOPT_POSTFIELDS(3)*.

While the **easy_handle** is added to a multi handle, it cannot be used by *curl_easy_perform(3)*.

EXAMPLE

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

```
    res = curl_easy_perform(curl);
    curl_easy_cleanup(curl);
  }
}
```

AVAILABILITY

Always

RETURN VALUE

CURLE_OK (0) means everything was OK, non-zero means an error occurred as <*curl/curl.h*> defines - see *libcurl-errors(3)*. If the *CURLOPT_ERRORBUFFER(3)* was set with *curl_easy_setopt(3)* there is a readable error message stored in the error buffer when non-zero is returned.

SEE ALSO

curl_easy_init(3), **curl_easy_setopt**(3), **curl_multi_add_handle**(3), **curl_multi_perform**(3), **libcurl-errors**(3)