How can I use CURLOPT_HEADERFUNCTION to read a single response header field?

Asked 7 years, 3 months ago Modified 7 years, 3 months ago Viewed 16k times

9

I'm implementing a C program which needs to read a remote file's size from the `Content-Length` header (when `Content-Length` is sent in the response headers).

I've looked over libcurl's docs, and the best I've been able to come up with so far has been a callback function for the `CURLOPT_HEADERFUNCTION` setting. I've put together a toy implementation of a callback, which is supposed to print the headers to `STDOUT` :

```
size_t hdf(char* b, size_t size, size_t nitems, void *userdata) {
    printf("%s", b);
    return 0;
}
```

While I want to be able to print the `Content-Length` header (or, at least, print all the headers), I can only get this function to print the response code:

```
$ ./curltest "some_url_which_sends_back_Content_Length"
HTTP/1.1 200 OK
```

If I comment out the line in my `main` which sets the callback to the `hdf` function defined above, the default behaviour is to print all the headers to `STDOUT` .

For reference, here is the `main` function I'm using, based on a thread on the libcurl mailing list:

```
int main(int argc, char *argv[])
{
   CURLcode ret;
   CURL *hnd = curl_easy_init();
   curl_easy_setopt(hnd, CURLOPT_URL, argv[1]);
   curl_easy_setopt(hnd, CURLOPT_HEADER, 1);
   curl_easy_setopt(hnd, CURLOPT_NOBODY, 1);
   curl_easy_setopt(hnd, CURLOPT_HEADERFUNCTION, hdf);
   ret = curl_easy_perform(hnd);
   curl_easy_cleanup(hnd);
}
```

How can I write a callback for the `CURLOPT_HEADERFUNCTION` option which can load a specific header into memory or otherwise manipulate it -- or, at least, load all headers into memory?

c libcurl

Jules
14.2k 13 56 101

1 Answer

Sorted by:
Highest score (default)

18

> I can only get this function to print the response code:

The answer to that is in the `CURLOPT_HEADERFUNCTION` documentation:

> This function gets called by libcurl as soon as it has received header data. The header callback will be called once for each header and only complete header lines are passed on to the callback. Parsing headers is very easy using this. **The size of the data pointed to by buffer is size multiplied with nmemb. Do not assume that the header line is zero terminated!** The pointer named userdata is the one you set with the CURLOPT_HEADERDATA option. **This callback function must return the number of bytes actually taken care of. If that amount differs from the amount passed in to your function, it'll signal an error to the library. This will cause the transfer to get aborted and the libcurl function in progress will return CURLE_WRITE_ERROR**.

Your `printf()` call is assuming null termination, and your callback is returning fewer bytes than provided, so you are aborting the response after the first response line is received.

Try this instead.

```
size_t hdf(char* b, size_t size, size_t nitems, void *userdata) {
    size_t numbytes = size * nitems;
    printf("%.*s\n", numbytes, b);
    return numbytes;
}
```

> How can I write a callback for the CURLOPT_HEADERFUNCTION option which can load a specific header into memory or otherwise manipulate it -- or, at least, load all headers into memory?

Once you fix your bug, you should be able to see all headers. You can then parse `b` looking for the `Content-Length` header, and when found save its data to a buffer that you pass to `userdata` via `CURLOPT_HEADERDATA`.

Now, with that said, there is an easier way to retrieve the Content-Length value. Perform a `HEAD` request with `curl_easy_perform()` (`CURLOPT_NOBODY`), and if successful than use `curl_easy_getinfo()` to retrieve the `CURLINFO_CONTENT_LENGTH_DOWNLOAD` value:

> CURLINFO_CONTENT_LENGTH_DOWNLOAD
> Content length from the Content-Length header.

Share Follow

edited May 30, 2016 at 2:52

answered May 30, 2016 at 2:40

Remy Lebeau
555k 31 458 770

1 D'oh, I've got to pay more attention to the docs. Thanks for the explanation! – Jules May 30, 2016 at 2:57