NAME

CURLOPT_PREREQFUNCTION − user callback called when a connection has been established, but before a request has been made.

SYNOPSIS

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

/* These are the return codes for the pre-request callback. */
#define CURL_PREREQFUNC_OK 0
#define CURL_PREREQFUNC_ABORT 1 /* fail the entire transfer */

int prereq_callback(void *clientp,
                    char *conn_primary_ip,
                    char *conn_local_ip,
                    int conn_primary_port,
                    int conn_local_port);

CURLcode curl_easy_setopt(CURL *handle, CURLOPT_PREREQFUNCTION, prereq_callback);
```

DESCRIPTION

Pass a pointer to your callback function, which should match the prototype shown above.

This function gets called by libcurl after a connection has been established or a connection has been reused (including any SSL handshaking), but before any request is actually made on the connection. For example, for HTTP, this callback is called once a connection has been established to the server, but before a GET/HEAD/POST/etc request has been sent.

This function may be called multiple times if redirections are enabled and are being followed (see *CURLOPT_FOLLOWLOCATION(3)*).

The callback function must return *CURL_PREREQFUNC_OK* on success, or *CURL_PREREQFUNC_ABORT* to cause the transfer to fail with result *CURLE_ABORTED_BY_CALLBACK*.

This function is passed the following arguments:

conn_primary_ip
: A pointer to a null−terminated C string containing the primary IP of the remote server established with this connection. For FTP, this is the IP for the control connection. IPv6 addresses are represented without surrounding brackets.

conn_local_ip
: A pointer to a null−terminated C string containing the originating IP for this connection. IPv6 addresses are represented without surrounding brackets.

conn_primary_port
: The primary port number on the remote server established with this connection. For FTP, this is the port for the control connection. This can be a TCP or a UDP port number depending on the protocol.

conn_local_port
: The originating port number for this connection. This can be a TCP or a UDP port number depending on the protocol.

clientp
: The pointer you set with *CURLOPT_PREREQDATA(3)*.

DEFAULT

NULL

PROTOCOLS

This functionality affects all supported protocols

EXAMPLE

```
struct priv {
  void *custom;
};

static int prereq_callback(void *clientp,
                           char *conn_primary_ip,
                           char *conn_local_ip,
                           int conn_primary_port,
                           int conn_local_port)
{
  printf("Connection made to %s:%d\n", conn_primary_ip, conn_primary_port);
  return CURL_PREREQFUNC_OK;
}

int main(void)
{
  struct priv prereq_data;
  CURL *curl = curl_easy_init();
  if(curl) {
    CURLcode result;
    curl_easy_setopt(curl, CURLOPT_PREREQFUNCTION, prereq_callback);
    curl_easy_setopt(curl, CURLOPT_PREREQDATA, &prereq_data);
    result = curl_easy_perform(curl);
    curl_easy_cleanup(curl);
  }
}
```

AVAILABILITY

Added in curl 7.80.0

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

CURLINFO_PRIMARY_IP(3), **CURLINFO_PRIMARY_PORT**(3), **CURLOPT_PREREQDATA**(3)