NAME

curl_ws_send − send WebSocket data

SYNOPSIS

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

CURLcode curl_ws_send(CURL *curl, const void *buffer, size_t buflen,
                      size_t *sent, curl_off_t fragsize,
                      unsigned int flags);
```

DESCRIPTION

Send the specific message chunk over an established WebSocket connection. *buffer* must point to a valid memory location containing (at least) *buflen* bytes of payload memory.

sent is set to the number of payload bytes actually sent. If the return value is **CURLE_OK** but *sent* is less than the given *buflen*, libcurl was unable to consume the complete payload in a single call. In this case the application must call this function again until all payload is processed. *buffer* and *buflen* must be updated on every following invocation to only point to the remaining piece of the payload.

fragsize should always be set to zero unless a (huge) frame shall be sent using multiple calls with partial content per call explicitly. In that case you must set the *CURLWS_OFFSET* bit and set the *fragsize* as documented in the section on *CURLWS_OFFSET* below.

flags must contain at least one flag indicating the type of the message. To send a fragmented message consisting of multiple frames, additionally set the *CURLWS_CONT* bit in all frames except the final one. The appropriate message type bit should be set in every frame of a fragmented message without exemption. Omitting the message type for continuation frames of a fragmented message is only supported for backwards compatibility and highly discouraged.

For more details on the supported flags see below and in *curl_ws_meta(3)*.

If *CURLWS_RAW_MODE* is enabled in *CURLOPT_WS_OPTIONS(3)*, the *flags* argument should be set to 0.

Warning: while it is possible to invoke this function from a callback, such a call is blocking in this situation, e.g. only returns after all data has been sent or an error is encountered.

FLAGS

Supports all flags documented in *curl_ws_meta(3)* and additionally the following flags.

CURLWS_OFFSET
: The provided data is only a partial frame and there is more coming in a following call to *curl_ws_send()*. When sending only a piece of the frame like this, the *fragsize* must be provided with the total expected frame size in the first call and must be zero in all subsequent calls.

PROTOCOLS

This functionality affects ws only

EXAMPLE

```
#include <string.h> /* for strlen */

int main(void)
{
  const char *buffer = "PAYLOAD";
  size_t offset = 0;
  CURLcode result = CURLE_OK;
  CURL *curl = curl_easy_init();
```

```
 curl_easy_setopt(curl, CURLOPT_URL, "wss://example.com/");
 curl_easy_setopt(curl, CURLOPT_CONNECT_ONLY, 2L);
 /* start HTTPS connection and upgrade to WSS, then return control */
 curl_easy_perform(curl);

 while(!result) {
  size_t sent;
  result = curl_ws_send(curl, buffer + offset,
                        strlen(buffer) - offset, &sent,
                        0, CURLWS_TEXT);
  offset += sent;

  if(result == CURLE_OK) {
    if(offset == strlen(buffer))
      break; /* finished sending */
  }

  if(result == CURLE_AGAIN)
    /* in real application: wait for socket here, e.g. using select() */
    result = CURLE_OK;
 }

 curl_easy_cleanup(curl);
 return (int)result;
}
```

AVAILABILITY

Added in curl 7.86.0

RETURN VALUE

This function returns a CURLcode indicating success or error.

CURLE_OK (0) means everything was OK, non−zero means an error occurred, see *libcurl−errors(3)*. If *CURLOPT_ERRORBUFFER(3)* was set with *curl_easy_setopt(3)* there can be an error message stored in the error buffer when non−zero is returned.

Instead of blocking, the function returns **CURLE_AGAIN**. The correct behavior is then to wait for the socket to signal readability before calling this function again.

Any other non−zero return value indicates an error. See the *libcurl−errors(3)* man page for the full list with descriptions.

SEE ALSO

curl_easy_getinfo(3), **curl_easy_perform**(3), **curl_easy_setopt**(3), **curl_ws_recv**(3), **curl_ws_start_frame**(3), **libcurl-ws**(3)