

NAME

curl_ws_recv - receive WebSocket data

SYNOPSIS

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

```
CURLcode curl_ws_recv(CURL *curl, void *buffer, size_t buflen,  
                      size_t *recv, const struct curl_ws_frame **meta);
```

DESCRIPTION

This function call is EXPERIMENTAL.

Retrieves as much as possible of a received WebSocket data fragment into the **buffer**, but not more than **buflen** bytes. *recv* is set to the number of bytes actually stored.

If there is more fragment data to deliver than what fits in the provided *buffer*, libcurl returns a full buffer and the application needs to call this function again to continue draining the buffer.

The *meta* pointer gets set to point to a *const struct curl_ws_frame* that contains information about the received data. See the *curl_ws_meta(3)* for details on that struct.

EXAMPLE

```
int main(void)  
{  
    size_t rlen;  
    const struct curl_ws_frame *meta;  
    char buffer[256];  
    CURL *curl = curl_easy_init();  
    if(curl) {  
        CURLcode res = curl_ws_recv(curl, buffer, sizeof(buffer), &rlen, &meta);  
        if(res)  
            printf("error: %s\n", curl_easy_strerror(res));  
    }  
}
```

AVAILABILITY

Added in 7.86.0.

RETURN VALUE

Returns **CURLE_OK** if everything is okay, and a non-zero number for errors. Returns

CURLE_GOT_NOTHING if the associated connection is closed.

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.

SEE ALSO

curl_easy_setopt(3), curl_easy_perform(3), curl_easy_getinfo(3), curl_ws_send(3), libcurl-ws(3)