

**NAME**

curl\_multi\_fdset - extracts file descriptor information from a multi handle

**SYNOPSIS**

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

```
CURLMcode curl_multi_fdset(CURLM *multi_handle,  
                           fd_set *read_fd_set,  
                           fd_set *write_fd_set,  
                           fd_set *exc_fd_set,  
                           int *max_fd);
```

**DESCRIPTION**

This function extracts file descriptor information from a given multi\_handle. libcurl returns its fd\_set sets. The application can use these to select() on, but be sure to FD\_ZERO them before calling this function as *curl\_multi\_fdset(3)* only adds its own descriptors, it doesn't zero or otherwise remove any others. The *curl\_multi\_perform(3)* function should be called as soon as one of them is ready to be read from or written to.

To be sure to have up-to-date results, you should call *curl\_multi\_perform* until it does not return CURLM\_CALL\_MULTI\_PERFORM prior to calling *curl\_multi\_fdset*. This will make sure that libcurl has updated the handles' socket states.

If no file descriptors are set by libcurl, *max\_fd* will contain -1 when this function returns. Otherwise it will contain the higher descriptor number libcurl set.

When doing select(), you should use **curl\_multi\_timeout** to figure out how long to wait for action. Call *curl\_multi\_perform* even if no activity has been seen on the fd\_sets after the timeout expires as otherwise internal retries and timeouts may not work as you'd think and want.

**RETURN VALUE**

CURLMcode type, general libcurl multi interface error code. See *libcurl-errors(3)*

**SEE ALSO**

**curl\_multi\_cleanup(3), curl\_multi\_init(3), curl\_multi\_timeout(3), curl\_multi\_perform(3)**