NAME

curl_multi_assign - set data to associate with an internal socket

SYNOPSIS

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

CURLMcode curl_multi_assign(CURLM *multi_handle, curl_socket_t sockfd,
                            void *sockptr);
```

DESCRIPTION

This function creates an association in the multi handle between the given socket and a private pointer of the application. This is designed for *curl_multi_socket_action(3)* uses.

When set, the *sockptr* pointer is passed to all future socket callbacks for the specific *sockfd* socket.

If the given *sockfd* is not already in use by libcurl, this function returns an error.

libcurl only keeps one single pointer associated with a socket, so calling this function several times for the same socket makes the last set pointer get used.

The idea here being that this association (socket to private pointer) is something that just about every application that uses this API needs and then libcurl can just as well do it since it already has the necessary functionality.

It is acceptable to call this function from your multi callback functions.

EXAMPLE

```
int main(void)
{
  CURLM *multi = curl_multi_init();
  void *ourstructp; /* pointer to our data */
  curl_socket_t fd; /* file descriptor to associate our data with */

  /* make our struct pointer associated with socket fd */
  CURLMcode mc = curl_multi_assign(multi, fd, ourstructp);
  if(mc)
    printf("error: %s\n", curl_multi_strerror(mc));
}
```

AVAILABILITY

Added in 7.15.5

RETURN VALUE

The standard CURLMcode for multi interface error codes.

TYPICAL USAGE

In a typical application you allocate a struct or at least use some kind of semi-dynamic data for each socket that we must wait for action on when using the *curl_multi_socket_action(3)* approach.

When our socket-callback gets called by libcurl and we get to know about yet another socket to wait for, we can use *curl_multi_assign(3)* to point out the particular data so that when we get updates about this same socket again, we do not have to find the struct associated with this socket by ourselves.

SEE ALSO

curl_multi_setopt(3), **curl_multi_socket_action**(3)