NAME
>   curl_global_init_mem - Global libcurl initialisation with memory callbacks

SYNOPSIS
>   **#include <curl/curl.h>**
>   **CURLcode curl_global_init_mem(long flags,**
>   **curl_malloc_callback m,**
>   **curl_free_callback f,**
>   **curl_realloc_callback r,**
>   **curl_strdup_callback s,**
>   **curl_calloc_callback c);**

DESCRIPTION
>   This function works exactly as *curl_global_init(3)* with one addition: it allows the application to set call-
>   backs to replace the otherwise used internal memory functions.
>
>   If you are using libcurl from multiple threads or libcurl was built with the threaded resolver option then the
>   callback functions must be thread safe. The threaded resolver is a common build option to enable (and in
>   some cases the default) so we strongly urge you to make your callback functions thread safe.
>
>   All callback arguments must be set to valid function pointers. The prototypes for the given callbacks must
>   match these:
>
>   void *malloc_callback(size_t size);
>   To replace malloc()
>
>   void free_callback(void *ptr);
>   To replace free()
>
>   void *realloc_callback(void *ptr, size_t size);
>   To replace realloc()
>
>   char *strdup_callback(const char *str);
>   To replace strdup()
>
>   void *calloc_callback(size_t nmemb, size_t size);
>   To replace calloc()
>   This function is otherwise the same as *curl_global_init(3)*, please refer to that man page for documentation.

CAUTION
>   Manipulating these gives considerable powers to the application to severely screw things up for libcurl.
>   Take care!

SEE ALSO
>   **curl_global_init**(3), **curl_global_cleanup**(3),