modbus\_mapping\_new(3)

=====================

NAME

----

modbus\_mapping\_new - allocate four arrays of bits and registers

SYNOPSIS

--------

\*modbus\_mapping\_t\* modbus\_mapping\_new(int 'nb\_bits', int 'nb\_input\_bits', int 'nb\_registers', int 'nb\_input\_registers');\*

DESCRIPTION

-----------

The \*modbus\_mapping\_new()\* function shall allocate four arrays to store bits,

input bits, registers and inputs registers. The pointers are stored in

modbus\_mapping\_t structure. All values of the arrays are initialized to zero.

This function is equivalent to a call of the

linkmb:modbus\_mapping\_new\_start\_address[3] function with all start addresses to

`0`.

If it isn't necessary to allocate an array for a specific type of data, you can

pass the zero value in argument, the associated pointer will be NULL.

This function is convenient to handle requests in a Modbus server/slave.

RETURN VALUE

------------

The function shall return the new allocated structure if successful. Otherwise

it shall return NULL and set errno.

ERRORS

------

\*ENOMEM\*::

Not enough memory

EXAMPLE

-------

[source,c]

-------------------

/\* The first value of each array is accessible from the 0 address. \*/

mb\_mapping = modbus\_mapping\_new(BITS\_ADDRESS + BITS\_NB,

INPUT\_BITS\_ADDRESS + INPUT\_BITS\_NB,

REGISTERS\_ADDRESS + REGISTERS\_NB,

INPUT\_REGISTERS\_ADDRESS + INPUT\_REGISTERS\_NB);

if (mb\_mapping == NULL) {

fprintf(stderr, "Failed to allocate the mapping: %s\n",

modbus\_strerror(errno));

modbus\_free(ctx);

return -1;

}

-------------------

SEE ALSO

--------

linkmb:modbus\_mapping\_free[3]

linkmb:modbus\_mapping\_new\_start\_address[3]

AUTHORS

-------

The libmodbus documentation was written by Stéphane Raimbault

<stephane.raimbault@gmail.com>