NAME

libbase64 - encode and decode base64

SYNOPSIS

#include <base64.h>

#define B64_ENCODE_LENGTH(size)
#define B64_DECODE_LENGTH(size)

size_t
b64_encode(*const char *src, size_t srcsz, char *dst, size_t dstsz*);

size_t
b64_decode(*const char *src, size_t srcsz, char *dst, size_t dstsz*);

DESCRIPTION

This library exposes few functions to encode and decode any kind of input data using base64 format.

The **B64_ENCODE_LENGTH**() and **B64_DECODE_LENGTH**() macros can be used to compute the required size to encode or decode respectively depending on the argument *size* expressed in bytes. They always return the number of bytes required at most (depending on padding bytes) but the returned value *doesn't* include the NUL terminator.

The **b64_encode**() function reads the input string *src* up to *srcsz* bytes (which can be -1 to read until NUL character) and stores the result into argument *dst*. The function writes up to *dstsz* bytes at most (including the NUL terminator).

The **b64_decode**() functions read the base64 encoded (or base64url encoded) input string *src* up to *srcsz* (which can be -1 to read until NUL character) and stores the decoded result into argument *dst*. Ths function writes up to *dstsz* bytes at most (including the NUL terminator). This function always append a NUL terminator so make sure to use the return value to get the real binary size if required.

RETURN VALUES

The functions **b64_encode**() and **b64_decode**() returns the number of bytes written into the *dst* argument (not including the NUL terminator) or (size_t)-1 on error. In that case *errno* is set to indicate the error.

ERRORS

[ERANGE] The output *dst* pointer wasn't large enough to store the encoded/decoded data.

[EILSEQ] Only set from **b64_decode**(). The input string *src* did not contain valid base64

characters.

AUTHORS

libbase64 was written by David Demelier <markand@malikania.fr>