NAME

libbuf - minimal dynamic string library

LIBRARY

libbuf (libbuf, -lbuf)

SYNOPSIS

#include <buf.h>

void
buf_clear(*struct buf *b*);

int
buf_dup(*struct buf *b, const struct buf *src*);

void
buf_erase(*struct buf *b, size_t pos, size_t count*);

void
buf_finish(*struct buf *b*);

int
buf_init(*struct buf *b*);

int
buf_printf(*struct buf *b, const char *fmt, ...*);

int
buf_putc(*struct buf *b, char c*);

int
buf_puts(*struct buf *b, const char *s*);

int
buf_reserve(*struct buf *b, size_t desired*);

int
buf_resize(*struct buf *b, size_t desired, char c*);

int

buf_shrink(*struct buf *b*);

int
buf_sub(*struct buf *b*, *const struct buf *src*, *size_t pos*, *size_t count*);

int
buf_vprintf(*struct buf *b*, *const char *fmt*, *va_list ap*);

DESCRIPTION

The **libbuf** library is a general purpose string library for C.

It automatically expands storage as required. It will first try to allocate twice as the current storage space until enough room is available or in case it would exceed maximum storage it will grow with minimal required storage.

Every function expects as first argument a buffer to work with which can never be NULL.

The *struct buf* is a publicly exposed structure with the following fields:

data *(char *)* The current NUL-terminated C string, maybe NULL if the buffer isn't initialized.

length *(size_t)* The data length.

capacity *(size_t)* The real capacity available for writing characters (not including NUL). A capacity of 5 means you can write 5 characters plus the NUL terminator without reallocating.

SEE ALSO

buf_clear(3), buf_dup(3), buf_erase(3), buf_finish(3), buf_init(3), buf_printf(3), buf_putc(3), buf_puts(3), buf_reserve(3), buf_resize(3), buf_shrink(3), buf_sub(3), buf_vprintf(3)

AUTHORS

The **libbuf** library was written by David Demelier <markand@malikania.fr>