Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'iconv.3' command

$ man iconv.3

ICONV(3) Linux Programmer's Manual ICONV(3)

NAME

iconv - perform character set conversion

SYNOPSIS

#include <iconv.h>

size_t iconv(iconv_t cd,

char **inbuf, size_t *inbytesleft,

char **outbuf, size_t *outbytesleft);

DESCRIPTION

The iconv() function converts a sequence of characters in one character encoding to a sequence of characters in another character encoding. The cd argument is a conversion descriptor, previously created by a call to iconv_open(3); the conversion descriptor defines the character encodings that iconv() uses for the conversion. The inbuf argument is the address of a variable that points to the first character of the in? put sequence; inbytesleft indicates the number of bytes in that buffer. The outbuf argument is the address of a variable that points to the first byte available in the output buffer; outbytesleft indicates the number of bytes available in the output buffer.

The main case is when inbuf is not NULL and *inbuf is not NULL. In this case, the iconv() function converts the multibyte sequence start? ing at *inbuf to a multibyte sequence starting at *outbuf. At most *inbytesleft bytes, starting at *inbuf, will be read. At most *out?

bytesleft bytes, starting at *outbuf, will be written.

The iconv() function converts one multibyte character at a time, and for each character conversion it increments *inbuf and decrements *in? bytesleft by the number of converted input bytes, it increments *outbuf and decrements *outbytesleft by the number of converted output bytes, and it updates the conversion state contained in cd. If the character encoding of the input is stateful, the iconv() function can also con? vert a sequence of input bytes to an update to the conversion state without producing any output bytes; such input is called a shift se? quence. The conversion can stop for four reasons:

1. An invalid multibyte sequence is encountered in the input. In this case, it sets errno to EILSEQ and returns (size_t) -1. *inbuf is left pointing to the beginning of the invalid multibyte sequence.
2. The input byte sequence has been entirely converted, that is, *in? bytesleft has gone down to 0. In this case, iconv() returns the number of nonreversible conversions performed during this call.
3. An incomplete multibyte sequence is encountered in the input, and the input byte sequence terminates after it. In this case, it sets errno to EINVAL and returns (size_t) -1. *inbuf is left pointing to the beginning of the incomplete multibyte sequence.
4. The output buffer has no more room for the next converted character. In this case, it sets errno to E2BIG and returns (size_t) -1.

A different case is when inbuf is NULL or *inbuf is NULL, but outbuf is not NULL and *outbuf is not NULL. In this case, the iconv() function attempts to set cd's conversion state to the initial state and store a corresponding shift sequence at *outbuf. At most *outbytesleft bytes, starting at *outbuf, will be written. If the output buffer has no more room for this reset sequence, it sets errno to E2BIG and returns (size_t) -1. Otherwise, it increments *outbuf and decrements *out? bytesleft by the number of bytes written.

A third case is when inbuf is NULL or *inbuf is NULL, and outbuf is NULL or *outbuf is NULL. In this case, the iconv() function sets cd's conversion state to the initial state.

RETURN VALUE

The iconv() function returns the number of characters converted in a nonreversible way during this call; reversible conversions are not counted. In case of error, it sets errno and returns (size_t) -1.

ERRORS

The following errors can occur, among others:

E2BIG There is not sufficient room at *outbuf.

EILSEQ An invalid multibyte sequence has been encountered in the input.

EINVAL An incomplete multibyte sequence has been encountered in the in? put.

VERSIONS

This function is available in glibc since version 2.1.

ATTRIBUTES

For an explanation of the terms used in this section, see at? tributes(7).

??

?Interface ? Attribute ? Value ?

??

?iconv() ? Thread safety ? MT-Safe race:cd ?

??

The iconv() function is MT-Safe, as long as callers arrange for mutual exclusion on the cd argument.

CONFORMING TO

POSIX.1-2001, POSIX.1-2008.

NOTES

In each series of calls to iconv(), the last should be one with inbuf or *inbuf equal to NULL, in order to flush out any partially converted input.

Although inbuf and outbuf are typed as char **, this does not mean that the objects they point can be interpreted as C strings or as arrays of characters: the interpretation of character byte sequences is handled internally by the conversion functions. In some encodings, a zero byte may be a valid part of a multibyte character.

The caller of iconv() must ensure that the pointers passed to the function are suitable for accessing characters in the appropriate character set. This includes ensuring correct alignment on platforms that have tight restrictions on alignment.

SEE ALSO

iconv_close(3), iconv_open(3), iconvconfig(8)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.