NAME

mbstowcs − convert a multibyte string to a wide-character string

LIBRARY

Standard C library (*libc*, *−lc*)

SYNOPSIS

#include <stdlib.h>

size_t mbstowcs(wchar_t *dest***[restrict .***n***], const char *restrict** *src***,**
size_t *n***);**

DESCRIPTION

If *dest* is not NULL, the **mbstowcs**() function converts the multibyte string *src* to a wide-character string starting at *dest*. At most*n* wide characters are written to *dest*. The sequence of characters in the strings*r c* shall begin in the initial shift state. The conversion can stop for three reasons:

- An invalid multibyte sequence has been encountered. In this case, *(size_t) −1* is returned.
- *n* non-L'\0' wide characters have been stored at *dest*. In this case, the number of wide characters written to *dest* is returned, but the shift state at this point is lost.
- The multibyte string has been completely converted, including the terminating null character ('\0'). In this case, the number of wide characters written to *dest*, excluding the terminating null wide character, is returned.

The programmer must ensure that there is room for at least *n* wide characters at *dest*.

If *dest* is NULL, *n* is ignored, and the conversion proceeds as above, except that the converted wide characters are not written out to memory, and that no length limit exists.

In order to avoid the case 2 above, the programmer should make sure *n* is greater than or equal to *mbstowcs(NULL,src,0)+1*.

RETURN VALUE

The **mbstowcs**() function returns the number of wide characters that make up the converted part of the wide-character string, not including the terminating null wide character. If an invalid multibyte sequence was encountered, *(size_t) −1* is returned.

ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

Interface	**Attribute**	**Value**
mbstowcs()	Thread safety	MT-Safe

STANDARDS

POSIX.1-2001, POSIX.1-2008, C99.

NOTES

The behavior of **mbstowcs**() depends on the **LC_CTYPE** category of the current locale.

The function **mbsrtowcs**(3) provides a better interface to the same functionality.

EXAMPLES

The program below illustrates the use of **mbstowcs**(), as well as some of the wide character classification functions. An example run is the following:

```
$ ./t_mbstowcs de_DE.UTF−8 Grüße!
Length of source string (excluding terminator):
    8 bytes
    6 multibyte characters

Wide character string is: Grüße! (6 characters)
    G alpha upper
```

```
            r alpha lower
            ü alpha lower
            ß alpha lower
            e alpha lower
            ! !alpha
```

Program source

```
    #include <locale.h>
    #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>
    #include <wchar.h>
    #include <wctype.h>

    int
    main(int argc, char *argv[])
    {
        size_t mbslen;      /* Number of multibyte characters in source */
        wchar_t *wcs;       /* Pointer to converted wide character string */

        if (argc < 3) {
            fprintf(stderr, "Usage: %s <locale> <string>\n", argv[0]);
            exit(EXIT_FAILURE);
        }

        /* Apply the specified locale. */

        if (setlocale(LC_ALL, argv[1]) == NULL) {
            perror("setlocale");
            exit(EXIT_FAILURE);
        }

        /* Calculate the length required to hold argv[2] converted to
           a wide character string. */

        mbslen = mbstowcs(NULL, argv[2], 0);
        if (mbslen == (size_t) −1) {
            perror("mbstowcs");
            exit(EXIT_FAILURE);
        }

        /* Describe the source string to the user. */

        printf("Length of source string (excluding terminator):\n");
        printf("    %zu bytes\n", strlen(argv[2]));
        printf("    %zu multibyte characters\n\n", mbslen);

        /* Allocate wide character string of the desired size.  Add 1
           to allow for terminating null wide character (L'\0'). */

        wcs = calloc(mbslen + 1, sizeof(*wcs));
        if (wcs == NULL) {
            perror("calloc");
            exit(EXIT_FAILURE);
        }

        /* Convert the multibyte character string in argv[2] to a
           wide character string. */

        if (mbstowcs(wcs, argv[2], mbslen + 1) == (size_t) −1) {
            perror("mbstowcs");
            exit(EXIT_FAILURE);
        }

        printf("Wide character string is: %ls (%zu characters)\n",
               wcs, mbslen);

        /* Now do some inspection of the classes of the characters in
           the wide character string. */

        for (wchar_t *wp = wcs; *wp != 0; wp++) {
            printf("    %lc ", (wint_t) *wp);

            if (!iswalpha(*wp))
                printf("!");
            printf("alpha ");

            if (iswalpha(*wp)) {
                if (iswupper(*wp))
                    printf("upper ");

                if (iswlower(*wp))
                    printf("lower ");
            }

            putchar('\n');
        }

        exit(EXIT_SUCCESS);
    }
```

SEE ALSO

mblen(3), **mbsrtowcs**(3), **mbtowc**(3), **wcstombs**(3), **wctomb**(3)