



Linux Ubuntu 26.04 Manual Pages on command 'duplocale.3'

\$ man duplocale.3

duplocale(3) Library Functions Manual duplocale(3)

NAME

duplocale - duplicate a locale object

LIBRARY

Standard C library (libc, -lc)

SYNOPSIS

```
#include <locale.h>
```

```
locale_t duplocale(locale_t locobj);
```

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

duplocale():

Since glibc 2.10:

```
_XOPEN_SOURCE >= 700
```

Before glibc 2.10:

```
_GNU_SOURCE
```

DESCRIPTION

The duplocale() function creates a duplicate of the locale object referred to by locobj.

If locobj is LC_GLOBAL_LOCALE, duplocale() creates a locale object containing a copy of the global locale determined by setlocale(3).

RETURN VALUE

On success, duplocale() returns a handle for the new locale object. On error, it returns (locale_t) 0, and sets errno to indicate the error.

ERRORS

ENOMEM Insufficient memory to create the duplicate locale object.

STANDARDS

POSIX.1-2008.

HISTORY

glibc 2.3.

NOTES

Duplicating a locale can serve the following purposes:

? To create a copy of a locale object in which one of more categories are to be modified (using `newlocale(3)`).

? To obtain a handle for the current locale which can be used in other functions that employ a locale handle, such as `toupper_l(3)`. This is done by applying `duplocale()` to the value returned by the following call:

```
loc = uselocale((locale_t) 0);
```

This technique is necessary, because the above `uselocale(3)` call may return the value `LC_GLOBAL_LOCALE`, which results in undefined behavior if passed to functions such as `toupper_l(3)`. Calling `duplocale()` can be used to ensure that the `LC_GLOBAL_LOCALE` value is converted into a usable locale object. See EXAMPLES, below.

Each locale object created by `duplocale()` should be deallocated using `freelocale(3)`.

EXAMPLES

The program below uses `uselocale(3)` and `duplocale()` to obtain a handle for the current locale which is then passed to `toupper_l(3)`. The program takes one command-line argument, a string of characters that is converted to uppercase and displayed on standard output. An example of its use is the following:

```
$ ./a.out abc;
```

```
ABC
```

Program source

```
#define _XOPEN_SOURCE 700

#include <ctype.h>
#include <err.h>
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int
main(int argc, char *argv[])
```

```

{
    locale_t loc, nloc;

    if (argc != 2) {
        fprintf(stderr, "Usage: %s string\n", argv[0]);
        exit(EXIT_FAILURE);
    }

    /* This sequence is necessary, because uselocale() might return
       the value LC_GLOBAL_LOCALE, which can't be passed as an
       argument to toupper_l(). */
    loc = uselocale((locale_t) 0);
    if (loc == (locale_t) 0)
        err(EXIT_FAILURE, "uselocale");
    nloc = duplocale(loc);
    if (nloc == (locale_t) 0)
        err(EXIT_FAILURE, "duplocale");
    for (char *p = argv[1]; *p; p++)
        putchar(toupper_l(*p, nloc));
    printf("\n");
    freelocale(nloc);
    exit(EXIT_SUCCESS);
}

```

SEE ALSO

freelocale(3), newlocale(3), setlocale(3), uselocale(3), locale(5), locale(7)