



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

\$ man pthread_cleanup_pop_restore_np.3

pthread_cleanup_push_defer_np(3) Library Functions Manual pthread_cleanup_push_defer_np(3)

NAME

pthread_cleanup_push_defer_np, pthread_cleanup_pop_restore_np - push and pop thread cancellation clean-up handlers while saving cancelability type

LIBRARY

POSIX threads library (libpthread, -lpthread)

SYNOPSIS

```
#include <pthread.h>

void pthread_cleanup_push_defer_np(typeof(void (void *)) *routine,
                                  void *arg);

void pthread_cleanup_pop_restore_np(int execute);
```

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

```
pthread_cleanup_push_defer_np(), pthread_cleanup_pop_defer_np():
    _GNU_SOURCE
```

DESCRIPTION

These functions are the same as pthread_cleanup_push(3) and pthread_cleanup_pop(3), except for the differences noted on this page.

Like pthread_cleanup_push(3), pthread_cleanup_push_defer_np() pushes routine onto the thread's stack of cancellation clean-up handlers. In addition, it also saves the thread's current cancelability type, and sets the cancelability type to "deferred" (see pthread_setcanceltype(3)); this ensures that cancellation clean-up will occur even if the thread's cancelability type was "asynchronous" before the call.

Like pthread_cleanup_pop(3), pthread_cleanup_pop_restore_np() pops the top-most clean-up

handler from the thread's stack of cancelation clean-up handlers. In addition, it restores the thread's cancelability type to its value at the time of the matching `pthread_cleanup_push_defer_np()`.

The caller must ensure that calls to these functions are paired within the same function, and at the same lexical nesting level. Other restrictions apply, as described in `pthread_cleanup_push(3)`.

This sequence of calls:

```
pthread_cleanup_push_defer_np(routine, arg);  
pthread_cleanup_pop_restore_np(execute);
```

is equivalent to (but shorter and more efficient than):

```
int oldtype;  
pthread_cleanup_push(routine, arg);  
pthread_setcanceltype(PTHREAD_CANCEL_DEFERRED, &oldtype);  
...  
pthread_setcanceltype(oldtype, NULL);  
pthread_cleanup_pop(execute);
```

STANDARDS

GNU; hence the suffix "_np" (nonportable) in the names.

HISTORY

glibc 2.0

SEE ALSO

`pthread_cancel(3)`, `pthread_cleanup_push(3)`, `pthread_setcancelstate(3)`, `pthread_testcancel(3)`, `pthread_cancel(3)`, `pthread_cancel(7)`