



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

### ***Rocky Enterprise Linux 9.2 Manual Pages on command 'on\_exit.3'***

**\$ man on\_exit.3**

ON\_EXIT(3)                      Linux Programmer's Manual                      ON\_EXIT(3)

NAME

on\_exit - register a function to be called at normal process termination

SYNOPSIS

```
#include <stdlib.h>
```

```
int on_exit(void (*function)(int , void *), void *arg);
```

Feature Test Macro Requirements for glibc (see feature\_test\_macros(7)):

on\_exit():

Since glibc 2.19:

```
    _DEFAULT_SOURCE
```

Glibc 2.19 and earlier:

```
    _BSD_SOURCE || _SVID_SOURCE
```

DESCRIPTION

The on\_exit() function registers the given function to be called at normal process termination, whether via exit(3) or via return from the program's main(). The function is passed the status argument given to the last call to exit(3) and the arg argument from on\_exit().

The same function may be registered multiple times: it is called once for each registration.

When a child process is created via fork(2), it inherits copies of its parent's registrations. Upon a successful call to one of the exec(3) functions, all registrations are removed.

RETURN VALUE

The `on_exit()` function returns the value 0 if successful; otherwise it returns a nonzero value.

## ATTRIBUTES

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

????????????????????????????????????????????????????????????

?Interface ? Attribute ? Value ?

????????????????????????????????????????????????????????????

?`on_exit()` ? Thread safety ? MT-Safe ?

????????????????????????????????????????????????????????????

## CONFORMING TO

This function comes from SunOS 4, but is also present in `glibc`. It no longer occurs in Solaris (SunOS 5). Portable application should avoid this function, and use the standard `atexit(3)` instead.

## NOTES

By the time function is executed, stack (auto) variables may already have gone out of scope. Therefore, `arg` should not be a pointer to a stack variable; it may however be a pointer to a heap variable or a global variable.

## SEE ALSO

`_exit(2)`, `atexit(3)`, `exit(3)`

## 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/>.