



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

\$ man atexit.3p

AEXIT(3P) POSIX Programmer's Manual AEXIT(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

atexit ? register a function to run at process termination

SYNOPSIS

```
#include <stdlib.h>

int atexit(void (*func)(void));
```

DESCRIPTION

The functionality described on this reference page is aligned with the ISO C standard. Any conflict between the requirements described here and the ISO C standard is unintentional. This volume of POSIX.1?2017 defers to the ISO C standard.

The `atexit()` function shall register the function pointed to by `func`, to be called without arguments at normal program termination. At normal program termination, all functions registered by the `atexit()` function shall be called, in the reverse order of their registration, except that a function is called after any previously registered functions that had already been called at the time it was registered. Normal termination occurs either by a call to `exit()` or a return from `main()`.

At least 32 functions can be registered with `atexit()`.

After a successful call to any of the `exec` functions, any functions previously registered by `atexit()` shall no longer be registered.

RETURN VALUE

Upon successful completion, `atexit()` shall return 0; otherwise, it shall return a non-zero value.

ERRORS

No errors are defined.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

The functions registered by a call to `atexit()` must return to ensure that all registered functions are called.

The application should call `sysconf()` to obtain the value of `{ATEXIT_MAX}`, the number of functions that can be registered. There is no way for an application to tell how many functions have already been registered with `atexit()`.

Since the behavior is undefined if the `exit()` function is called more than once, portable applications calling `atexit()` must ensure that the `exit()` function is not called at normal process termination when all functions registered by the `atexit()` function are called.

All functions registered by the `atexit()` function are called at normal process termination, which occurs by a call to the `exit()` function or a return from `main()` or on the last thread termination, when the behavior is as if the implementation called `exit()` with a zero argument at thread termination time.

If, at normal process termination, a function registered by the `atexit()` function is called and a portable application needs to stop further `exit()` processing, it must call the `_exit()` function or the `_Exit()` function or one of the functions which cause abnormal process termination.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

`exec`, `exit()`, `sysconf()`

The Base Definitions volume of POSIX.1-2017, `<stdlib.h>`

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.kernel.org/doc/man-pages/reporting_bugs.html.

IEEE/The Open Group

2017

AEXIT(3P)