

**NAME**

atexit - register a function to be called at normal process termination

**SYNOPSIS**

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

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

**DESCRIPTION**

The **atexit()** function registers the given *function* to be called at normal process termination, either via **exit(3)** or via return from the program's *main()*. Functions so registered are called in the reverse order of their registration; no arguments are passed.

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

POSIX.1-2001 requires that an implementation allow at least **ATEXIT\_MAX** (32) such functions to be registered. The actual limit supported by an implementation can be obtained using **sysconf(3)**.

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 **atexit()** function returns the value 0 if successful; otherwise it returns a nonzero value.

**CONFORMING TO**

SVr4, 4.3BSD, C89, C99, POSIX.1-2001.

**NOTES**

Functions registered using **atexit()** (and **on\_exit(3)**) are not called if a process terminates abnormally because of the delivery of a signal.

If one of the functions registered functions calls **\_exit(2)**, then any remaining functions are not invoked, and the other process termination steps performed by **exit(3)** are not performed.

POSIX.1-2001 says that the result of calling **exit(3)** more than once (i.e., calling **exit(3)** within a function registered using **atexit()**) is undefined. On some systems (but not Linux), this can result in an infinite recursion; portable programs should not invoke **exit(3)** inside a function registered using **atexit()**.

The **atexit()** and **on\_exit(3)** functions register functions on the same list: at normal process termination, the registered functions are invoked in reverse order of their registration by these two functions.

POSIX.1-2001 says that the result is undefined if **longjmp(3)** is used to terminate execution of one of the functions registered **atexit()**.

**Linux notes**

Since glibc 2.2.3, **atexit()** (and **on\_exit(3)**) can be used within a shared library to establish functions that are called when the shared library is unloaded.

**EXAMPLE**

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

void
bye(void)
{
    printf("That was all, folks\n");
}

int
main(void)
{
    long a;
    int i;
```

```
a = sysconf(_SC_ATEXIT_MAX);
printf("ATEXIT_MAX = %ld\n", a);

i = atexit(bye);
if (i != 0) {
    fprintf(stderr, "cannot set exit function\n");
    exit(EXIT_FAILURE);
}

exit(EXIT_SUCCESS);
}
```

**SEE ALSO**

[\\_exit\(2\)](#), [exit\(3\)](#), [on\\_exit\(3\)](#)

**COLOPHON**

This page is part of release 3.74 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 <http://www.kernel.org/doc/man-pages/>.