NAME

close - delete a descriptor

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <unistd.h>

int
close(*int fd*);

DESCRIPTION

The **close**() system call deletes a descriptor from the per-process object reference table. If this is the last reference to the underlying object, the object will be deactivated. For example, on the last close of a file the current *seek* pointer associated with the file is lost; on the last close of a socket(2) associated naming information and queued data are discarded; on the last close of a file holding an advisory lock the lock is released (see further flock(2)). However, the semantics of System V and IEEE Std 1003.1-1988 ("POSIX.1") dictate that all fcntl(2) advisory record locks associated with a file for a given process are removed when *any* file descriptor for that file is closed by that process.

When a process exits, all associated file descriptors are freed, but since there is a limit on active descriptors per processes, the **close**() system call is useful when a large quantity of file descriptors are being handled.

When a process forks (see fork(2)), all descriptors for the new child process reference the same objects as they did in the parent before the fork. If a new process is then to be run using execve(2), the process would normally inherit these descriptors. Most of the descriptors can be rearranged with dup2(2) or deleted with **close**() before the execve(2) is attempted, but if some of these descriptors will still be needed if the execve fails, it is necessary to arrange for them to be closed if the execve succeeds. For this reason, the call "fcntl(d, F_SETFD, FD_CLOEXEC)" is provided, which arranges that a descriptor will be closed after a successful execve; the call "fcntl(d, F_SETFD, 0)" restores the default, which is to not close the descriptor.

RETURN VALUES

The **close**() function returns the value 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

The **close**() system call will fail if:

[EBADF] The *fd* argument is not an active descriptor.

[EINTR] An interrupt was received.

[ENOSPC] The underlying object did not fit, cached data was lost.

[ECONNRESET] The underlying object was a stream socket that was shut down by the peer before all pending data was delivered.

In case of any error except EBADF, the supplied file descriptor is deallocated and therefore is no longer valid.

SEE ALSO

accept(2), closefrom(2), execve(2), fcntl(2), flock(2), open(2), pipe(2), socket(2), socketpair(2)

STANDARDS

The **close**() system call is expected to conform to IEEE Std 1003.1-1990 ("POSIX.1").

HISTORY

The **close**() function appeared in Version 1 AT&T UNIX.